



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



ЗБОРНИК РАДОВА ФАКУЛТЕТА ТЕХНИЧКИХ НАУКА

Едиција: Техничке науке - зборници

Година: XXXIV

Број: 11/2019

Нови Сад

Едиција: „Техничке науке – Зборници“

Година: XXXIV

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 31.05.2019. до 20.09.2019. год., а који се промовишу 19.12.2019. 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 и
- електротехнике и рачунарства.

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

- грађевинарства,
- саобраћаја,
- графичког инжењерства и дизајна,
- архитектуре,
- инжењерског менаџмента.
- инжењерства заштите на раду и заштите животне средине,
- математике у техници,
- геодезије и геоматике,
- регионалне политике и развоја,
- инжењерства информационих система,
- сценског дизајна,
- биомедицинског инжењерства,
- информационог инжењерства и
- чистих енергетских технологија.

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. Милан Радовановић, Себастиан Балаш, УТИЦАЈ ПАРАМЕТАРА РОБОТСКОГ НАВАРИВАЊА НА ОСОБИНЕ ИЗРАЂЕНОГ ЕЛЕМЕНТА КОНСТРУКЦИЈЕ ОД ЧЕЛИКА S355	1925-1928
2. Драган Перановић, Себастиан Балаш, ЕКСПЕРИМЕНТАЛНО ОДРЕЂИВАЊЕ УТИЦАЈА ОКСИДНИХ ТОПИТЕЉА НА ТИП ПОСТУПАК ЗАВАРИВАЊА	1929-1932
3. Zvonko Magoč, Mladomir Milutinović, RAZVOJ IDEJNOG REŠENJA I KONSTRUKCIJE ALATA ZA IZRADU FITINGA LUKOVA Ø110/15° Ø110/30°.....	1933-1936
4. Vanja Vidović, Mladomir Milutinović, POLIMERNI MATERIJALI U MEDICINI I FARMACIJI I POSTUPCI NJIHOVE PRERADE	1937-1940
5. Ana Vujinov, Borislav Savković, IZBOR MERODAVNIH REŽIMA OBRADE PRI IZRADI KUĆIŠTA VENTILA ITRON	1941-1944
6. Nikola Đokić, Đorđe Vukelić, TRIBODIJAGNOSTIKA STATIČKOG KOEFICIJENTA TRENJA KONTAKTNIH PAROVA OD DRVETA	1945-1948
7. Nikola Vranješević, ISTRAŽIVANJE STRUJANJA KROZ CEV MEŠAVINE VODE I ETILEN GLIKOLA SA NANOČESTICAMA SREBRA	1949-1952
8. Dalibor Pelagić, ANALIZA TRANZIJENTNIH KARAKTERISTIKA BOČNE DINAMIKE VOZILA POMOĆU SOFTVERA ZA MODELIRANJE I SIMULACIJE	1953-1956

Radovi iz oblasti: Elektrotehnika i računarstvo

1. Marko Knežević, Dragan Pejić, SIMULACIONI MODEL METODE ZA GENERISANJE ŠUMA DISKRETNOG U VREMENU I KONTINUALNOG PO VREDNOSTI	1957-1960
--	-----------

2. Dragana Stajić, Vladimir Katić, OBNOVLJIVI IZVORI ENERGIJE U ZAŠTIĆENIM PODRUČJIMA I POTENCIJAL KORIŠĆENJA SOLARNE ENERGIJE U MANASTIRU KOVILJ	1961-1964
3. Aleksandar Ljahović, APLIKACIJA ZA PRAĆENJE CENA PO MARKETIMA	1965-1968
4. Стефана Томић, CLOSURES IN JAVASCRIPT – ДЕТАЉНО ОБЈАШЊЕЊЕ И ПРИМЕНА	1969-1972
5. Jelena Ljubišić, RAZVOJ INFORMACIONOG SISTEMA ZA AUTO-ŠKOLE	1973-1976
6. Stefan Dupljanin, Marko Vekić, RAZVOJ MODELA ELEKTRIČNOG AUTOMOBILA U MATLAB SIMULINKU	1977-1980
7. Igor Milanović, Milan Vidaković, INSTANA ALAT ZA NADGLEDANJE PERFORMANSI APLIKACIJA I JAVA ATTACH API	1981-1984
8. Stevan Mihajlović, Vladimir Katić, Octavian Cornea, MODELOVANJE ENERGETSKIH PRETVARAČA U MIKROMREŽAMA.....	1985-1988
9. Vukan Ninković, Dejan Vukobratović, Dejan Nemec LOKALIZACIJA MOBILNIH UREĐAJA U ZATVORENOM PROSTORU POMOĆU MAŠINSKOG UČENJA	1989-1992
10. Jovana Janković, Vladimir Katić UTICAJ PROPADA NAPONA NA RAD PUNIONICA ZA ELEKTRIČNA VOZILA NAPAJANIH IZ SISTEMA KROVNIH FN ELEKTRANA	1993-1996
11. Mihajlo Govorčin, Marko Vekić, MODEL STATIČKOG VAR KOMPENZATORA PRI REGULACIJI NAPONA.....	1997-2000
12. Vladislav Simić, GENERISANJE KODA ZA VIZUALIZACIJU REZULTATA ELEKTROENERGETSKIH PRORAČUNA UZ UPOTREBU OBLASNO SPECIFIČNOG JEZIKA	2001-2004
13. Светлана Ђурић, ИМПЛЕМЕНТАЦИЈА И ВИЗУАЛИЗАЦИЈА ГРАФА ТОКА КОНТРОЛЕ НА ЈЕЗИКУ PHARO	2005-2008
14. Aleksandar Ratkov, DIZAJN SERVERLESS WEB APLIKACIJA NA AMAZON PLATFORMI	2009-2011
15. Dejan Grubišić, SISTEM ZA PRONALAŽENJE NAJKRAĆE PUTANJE IZMEĐU LOKACIJA SA TEŽINAMA PUTA- NJA PROMENLJIVIM U REALNOM VREMENU ZASNOVAN NA ARHITEKTURI LAMBDA.....	2012-2015
16. Maja Milišić, Željen Trpovski, METODE ISPITIVANJA I GARANCIJE KVALITETA U OBLASTI SOFTVERSKIH ALATA	2016-2019
17. Luka Janičić, Boris Dumnić, UPRAVLJANJE SAVREMENIM KOMPRESORSKM SISTEMIMA ZA TRANSPORT INDUSTRIJSKIH GASOVA	2020-2023
18. Nikola Jerković, KRATKOROČNA PROGNOZA REŽIMA DISTRIBUTIVNE MREŽE	2024-2027
19. Andor Rajšli, Vladimir Katić, MODELOVANJE MALE HIDROELEKTRANE NA BRANI KOD NOVOG BEČEJA	2028-2031
20. Luka Ivezić, PRIMENA METODA JEDNOSTAVNOG EKSPONENCIJALNOG IZGLAĐIVANJA ZA PRORAČUN PROGNOZE PROTOKA PRIRODNOG GASA	2032-2035
21. Jovana Marković, USREDNJNI MODEL FN INVERTORA SA NAPREDNIM UPRAVLJANJEM	2036-2039
22. Anja Subotin, Vladimir Katić, PREGLED SOFTVERSKIH ALATA ZA SIMULACIJU I ODABIR PARAMETARA TOPLOTNIH PUMPI	2040-2043

23. Nikola Gavrić, MODEL DDOS NAPADA APLIKACIONOG SLOJA GAMING SERVERA EKSPLOATACIJOM KLIJENTSKE APLIKACIJE	2044-2047
24. Stefan Đorđević, VELIKI PODACI U ELEKTROENERGETSKIM SISTEMIMA	2048-2051
25. Nebojša Horvat, RAZVOJ I UPOREDNA ANALIZA PROGRAMSKIH JEZIKA ZA PAMETNE UGOVORE NA BLOKČEJNU	2052-2055
26. Ања Буљевић, АДАПТИВНО УПРАВЉАЊЕ КРАНОМ ПРИМЈЕНОМ „GAIN-SCHEDULING " АЛГОРИТМА	2056-2059
27. Aleksandar Mitrović, FAZNO SPREGNUTA PETLJA I PROCENA NEURAVNOTEŽENOSTI MREŽE PRIMENOM VIŠEREZONANTNOG OPSERVERA	2060-2063
28. Jovana Kovačević, HIBRIDNI ESTIMATOR STANJA SA STRUJAMA U PRAVOUGAONIM KOORDINATAMA ZA UKLJUČIVANJE FAZORSKIH MERENJA	2064-2067
29. Bojan Stipić, Zorica Suvajdžin Rakić, PROTOKOL JEZIČKIH SERVERA	2068-2071
30. Никола Ђуза, АУТОМАТСКО ТЕСТИРАЊЕ МИКРО СЕРВИСА АЛАТИМА ЗА ТЕСТИРАЊЕ ПРИХВАТЉИВОСТИ	2072-2075
31. Aleksa Mitreski, PC APLIKACIJA ZA EDUKATIVNI ELEKTROFIZIOLOŠKI SISTEM	2076-2079
32. Miloš Kudus, PREDVIĐANJE EFEKATA TEMPORALNOG FILTRA NA KVALITET VIDEO SEKVENCI	2080-2083
33. Milorad Vojnović, JEZIK ZA GENERISANJE API TESTOVA ZA GRAPHQL UPITE	2084-2087
34. Milica Nikolić, SISTEM ZA UPRAVLJANJE ŽALBAMA KORISNIKA U DOMENU FINANSIJA	2088-2091
35. Milica Delić, Luka Strezoski, POREĐENJE PERFORMANSI MATRIČNOG I ALGORITMA SUMIRANJA STRUJA ZA PRORAČUN REŽIMA DISTRIBUTIVNE MREŽE SA KRATKIM SPOJEM	2092-2095
36. Milana Komosar, Michael Rock, Vladimir Katić, MODEL I SIMULACIJE VETROGENERATORA U ATP/EMTP-U I MATLAB/SIMULINK-U ZA SLUČAJ ATMOSFERSKOG PRAŽNJENJA	2096-2099
37. Aleksandra Stevanović, Vladimir Katić, BEZBEDONOSNI I INFORMACIONI SISTEM SOLARNIH ELEKTRANA SA PRORAČUNOM OSVETLJENJA	2100-2103
38. Milena Lisov, MODELOVANJE KORELACIJE MERENJA U ESTIMACIJI STANJA	2104-2107
39. Марина Радовановић, NIGHTMARE MAGIC – ПРИМЕР УПОТРЕБЕ ЕЛЕКТРОЕНЦЕФАЛОГРАФИЈЕ И ЕМОТИВ ЕРОС УРЕЂАЈА У РАЧУНАРСКИМ ИГРАМА	2108-2111
40. Olja Anđelovski, Srđan Popov, OBJEKTNA PARADIGMA RUBY PROGRAMSKOG JEZIKA SA KOMPARATIVNOM ANALIZOM RUBY ON RAILS I JAVA SPRING OKRUŽENJA.....	2112-2115

УТИЦАЈ ПАРАМЕТАРА РОБОТСКОГ НАВАРИВАЊА НА ОСОБИНЕ ИЗРАЂЕНОГ ЕЛЕМЕНТА КОНСТРУКЦИЈЕ ОД ЧЕЛИКА S355**INFLUENCE OF ROBOTIC HARDFACING PARAMETERS ON PROPERTIES OF THE S355 STRUCTURAL STEEL S355 FABRICATED ELEMENT**Милан Радовановић, Себастиан Балош, *Факултет техничких наука, Нови Сад***Област – МАШИНТВО**

Кратак садржај - У раду је извршено експериментално наваривање додатног материјала SG2 слој по слој на плочу од конструкционог челика уз помоћ робота. Израђена су три узорка са различитим параметрима и смеровима наваривања. Добијени наварени слојеви су визуелно испитани, испитани су макро пресеци, а извршено је и испитивање затезањем, савијањем и енергија удара. Испитивањем је утврђено да параметри као што су мања струја и напон уз оптималну брзину наваривања као и наизменичног смера наваривања значајно утичу на механичке особине материјала и геометрију добијеног елемента.

Кључне речи: роботско заваривање, MAG наваривање, механичке особине, WAAM технологија.

Abstract: In this work, experimental welding of the consumable material SG2 layer per layer on a plate of structural steel with the aid of robots was carried out. Three samples were created with different parameters and welding directions. The obtained welded joints were visually tested, macro cross-sections were examined, and tensile testing, bending, and impact energy tests were performed. The experiment found that parameters such as lower current and voltage with the optimal welding speed and the alternating welding direction significantly influence the mechanical properties of the material and geometry of the material.

Keywords: robot welding, GMAW hardfacing, mechanical properties, WAAM technology

1. УВОД

Заваривање је технологија спајања два или више, истих или различитих материјала, топљењем или притиском, са или без додавања додатног материјала. Поступци заваривања, који се најчешће користе у пракси, засновани су на локалном загревању материјала изнад температуре топљења, када заварени спој настаје очвршћавањем (електролучно), или на локалном загревању заварени спој настаје уз додатно деловање притиска [1].

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији ментор је био др Себастиан Балош, ванр. проф.

Наваривање је процес наношења површинског слоја на метални део да би се повећала његова отпорност према абразији, ерозији, ударцима или неком другом облику оптерећења. Процес наваривања је у практичном смислу исти као заваривање, с тим да је његова намена другачија. Основна разлика у односу на метализацију је топљење површине основног материјала и његово мешање са додатним материјалом. За наваривање се највише користи Е, МИГ И ТИГ поступак, а кад је потребно нанети слој на што већу површину примењује се ЕПП или експлозивни поступак [2]. Предмет разматрања овог рада је нова технологија наваривања WAAM-Адитивна производња електричним луком и жицом. WAAM је једна од технологија са највећом брзином наношења материјала, знатно већом у односу на селективно ласерско синтеровање, при чему је цена опреме значајно нижа. Заснива се на наваривању слоја на претходни слој помоћу робота са извором за заваривање и додавачем жице. Користи се за наваривање великих, скувих металних делова са сложеном геометријом у појединачној производњи или у малим серијама, при чему је једино ограничење у величини производа радни простор робота.

2. ЕКСПЕРИМЕНТАЛНИ РАД

За наваривање као основни материјал коришћене су плоче од конструкционог челика S355J2+N дебљине 8 мм, димензија 400 мм × 100 мм, а као додатни материјал жица SG2 произвођача Thyssen energomag, пречника 1,2 мм у заштитном гасу C18 (Ar 82% CO₂ 18%).

Табела 1. Називни хемијски састав жице SG2 (EN ISO 14341-A-G 42 4 C1/M21 3Si1)

%C	%Si	%Mn	%P	%S	%Cu	%Cr	%Ni	%Ti
0,07	0,85	1,45	0,02	0,02	0,25	<0,15	<0,15	< 0,05

Табела 2. Механичка својства наноса жицом SG2 (EN ISO 14341-A-G 42 4 C1/M21 3Si1)

Затезна чврстоћа Rm [MPa]	Напон течења Re [MPa]	Издужење A [%]
550	450	>24

Поступак израде узорака спроведен је на роботској ћелији FANUC Robot ARC Mate 100iC/8L у фирми Proficut у Бачком Петровцу. За извор струје за наваривање коришћен је уређај EWM Phoenix puls 500A са воденим хлађењем.

Димензије узорака су 200мм × 80мм × 5-10 мм. Први узорак је рађен са путањом горионика само у једном смеру и увек са исте тачке у којој се успостављао електрични лук. Путање пиштоља за наваривање преостала два узорка биле су наизменичне, тј. први пролаз с лева у десно, следећи с десна у лево, и тако наизменично.

Током израде зидова, односно приликом наваривања сваког слоја читавањем су параметри наваривања (јачина струје и напон).

Након наваривања сваког пролаза помичним мерилем измерена је висина зида и очишћена је горња површина са паузама између сваког пролаза око 2-3 мин. Код израде првог зида (узорак 1) коришћен је гас Ar 82% i CO₂ 18% . Брзина заваривања је била константна и износила је 7 mm/sec. Слободни крај жице је 14 mm, а проток гаса 15 l/min. Остали параметри заваривања, као што су: јачина струје, напон , висина сваког пролаза и унесена топлота, приказани су у табели 3.

Табела 3. Параметри наваривања првог зида

Пролаз	Висина, [mm]	Јачина струје, [A]	Напон, [V]	Брзина жице [m/min]	Унесена топлота, [kJ/cm]
1.	3	160	23,9	6	4,37
2.	5	150	23,4	5,5	4,01
3.	6,7	140	23	5	3,68
4.	8,2	131	22,5	4,5	3,36
5.	9,8	121	22,1	4	3,05
6.	11,3	107	21,3	3,5	2,60
50.	77,3	107	21,3	3,5	2,60
51.	78,5	107	21,3	3,5	2,60
52.	80	107	21,3	3,5	2,60

Код израде другог зида (узорак 2) коришћен је гас Ar 82% CO₂ 18%. Брзина заваривања је константна и износи 10 mm/sec. Слободни крај жице је 15 mm, а проток плина 15 l/min. Остали параметри заваривања приказани су у табели 4.

Табела 4. Параметри наваривања другог зида

Пролаз	Висина, [mm]	Јачина струје, [A]	Напон, [V]	Брзина жице [m/min]	Унесена топлота, [kJ/cm]
1.	2,5	160	23,9	6	3,05
2.	4,4	160	23,9	6	3,05
3.	5,6	160	23,9	6	3,05
4.	6,8	160	23,9	6	3,05
5.	8	160	23,9	6	3,05
54.	78,5	160	23,9	6	3,05
55.	80	160	23,9	6	3,05
56.	81	160	23,9	6	3,05

Код израде трећег зида (узорак 3) коришћен је исти гас. Брзина заваривања је константна и износи 13 mm/sec. Слободни крај жице је 15 mm, а проток плина 15 l/min. Остали параметри заваривања приказани су у табели 5.

Табела 5. Параметри наваривања трећег зида

Пролаз	Висина, [mm]	Јачина струје, [A]	Напон, [V]	Брзина жице [m/min]	Унесена топлота, [kJ/cm]
1.	2,5	184	23,7	7	2,68
2.	4,3	184	23,7	7	2,68
3.	5,8	184	23,7	7	2,68
4.	7	184	23,7	7	2,68
5.	8,4	184	23,7	7	2,68
59.	79,5	184	23,7	7	2,68
60.	80,9	184	23,7	7	2,68
61.	81,6	184	23,7	7	2,68

Мора се напоменути да су сва три узорка рађена на температури око 0° целзијуса, као и да се хала која није имала грејање периодично отвара и проветрала током саме израде узорака, па се доводи упитање утицаја промаје, повећане влажности као и смрзавање гаса у боци. Након наваривања извршено је испитивање узорака у Лабораторији за испитивање материјала на Факултету техничких наука у Новом Саду. Поступак испитивања је урађен у складу са стандардом SRPS EN ISO 15614. Извршено је макро испитивање, испитивање затезањем, савијањем и енергија удара. Поступак испитивања узорака на затезање и на савијање је урађен на једноосној машини за затезање тј. кидалици типа ВЕБ ЗДМ 5/91. Испитивање жилавости је рађено на инструментаном Шарпијевом клатну JWT-450 у радном опсегу 450 J. При испитивању макро структуре у оба случаја извршено је исецање узорака са брусилцом, након тога извршено је постепено брушење са шмирглом класе р80 па до р2500, након тога је извршено полирање узорака и нагризање које је рађено са 3% ниталом (3% азотна киселина HNO₃ у алкохолу).

3. РЕЗУЛТАТИ

3.1 Резултати макро испитивања

Макро анализа рађена је на сва три узорка што се може видети са слике 1.



а)

б)

в)

Слика 1. Макро испитивање: а) узорак 1, б) узорак 2, в) узорак 3

Прегледом макроструктуре уочена је нагнутост на сва три узорка, а посебно се истиче на узорку 2 и 3. На узорку 1 се види равномерније слагање слојева у односу на друга два узорка, док је на месту завршетка лука најмањи прираштај висине што се може видети са слике а) леви узорак.

На последњем пролазу код сва три узорка могу јасно видети стубасти кристали. Видљива је порозност на сва три узорка а највећа је уочена на узорку 3.

3.2 Резултати испитивања затезањем

Резултати испитивања узорка на затезање приказани су у табели 6. На основу резултата може да се констатује да је највећа добијена вредност напона течења и затезне чврстоће постигнута на узорку 2 као и средња вредност која одговара прописаном стандарду EN10025. Најнижа вредност напона течења и затезне чврстоће добијена је на узорку 3.

Код свих узорка који су испитивани затезањем дошло је до лома епрувете на различитим местима, уз претходно формирање врата на епрувети насталог услед деформације затезањем. На слици 2 приказана је епрувета и изглед лома након испитивања узорка затезањем.

деловању силе унутрашњи напони немају довољну јачину да се супроставе деформацији и долази до отварања.



Слика 2. Изглед епрувете након испитивања затезањем: а) узорак 1, б) узорак 2, в) узорак 3

Табела 6. Резултати испитивања затезањем

Узорак	Епрувета	R_p [MPa]	R_m [MPa]	A [%]
1	11	439	484	28
	13	455	518	30
	17	402	500	32
	Ср. вредност	432	501	30
2	21	488	536	26
	23	465	543	30
	27	466	540	28
	Ср. вредност	473	539	28
3	31	242	252	6
	33	396	443	35
	37	451	473	12
	Ср. вредност	363	389	18
Механичке особине челика S355J2+N по стандарду EN10025		355-550	490-630	>22

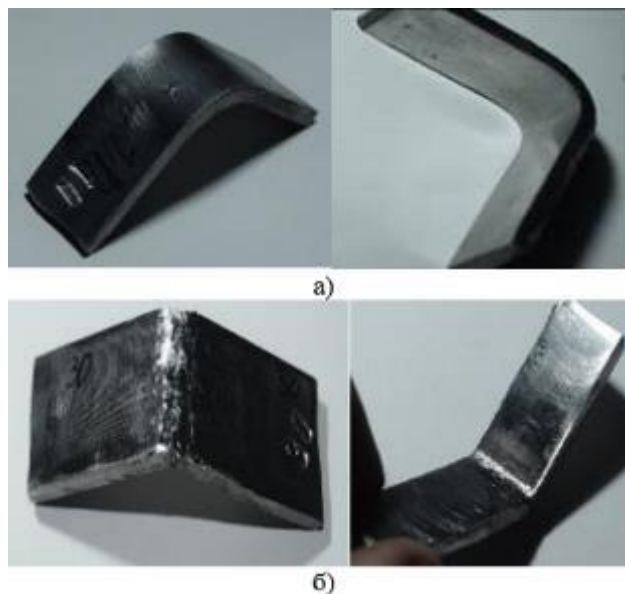
3.3 Резултати испитивања савијањем

Резултати испитивања видљиви су на узорцима са слике 3. где су приказана два узорка један најбољи и један најлошији. На основу резултата датих у табели 7. може да се констатује да код узорка 1 испитивањем савијањем преко лица није дошло до појаве прслине тј. отварања епрувете, док је код узорка 3 дошло до отварања.

Остали узорци из групе узорка 1 су имали чисту површину без прслина. Док код узорка из групе 3 постоји велика порозност код свих узорка па при

Табела 7. Резултати испитаних узорка на савијање

Узорак 1	Прслина мања од 2 мм
Узорак 2	Четири прслине мање од 1 мм са спољне стране, шест мањих од 2 мм са унутрашње
Узорка 3	Велика порозност на свим узорцима

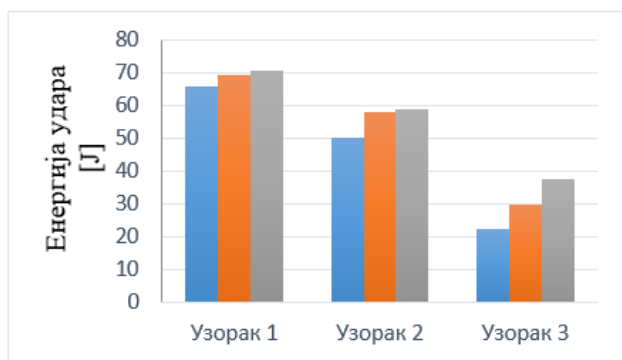


Слика 3. Узорци након испитивања на савијање: а) узорак 1, б) узорак 3

3.4 Резултати испитивања енергије удара

Са слике се види да узорак 1 у односу на друга два узорка поседује веће вредности жилавости материјала.

Мале вредности узорка 3 можемо преписати порозној структури која је последица спољашњих фактора и нешто већим вредностима параметара заваривања.



Слика 4. Вредности енергије удара за сва три узорка



Слика 5. Узорци након испитивања на Шарпију

4. ДИСКУСИЈА

При испитивању узорка на затезање израчунате су средње вредности за сва три узорка и упоређене са стандардним прописаним вредностима за конвенционално произведен челик основе на коју је извршено наваривање (ваљане плоче), тако да се узорци 1 и 2 налазе у прописаним вредностима за R_p , R_m и A , док се код узорка 3 једино напон течења налази у прописаним вредностима.

Током испитивања на савијање најмања прслина се јавила на узорку 1 док код других узорка добијених из узорка 1 нису јавиле прслине. На свим узорцима из узорка 3 се јасно види порозност што је довело до отварања материјала. Такође се јасно види да је узорак 3 има најмање вредности при испитивању на енергију удара, што значи да је жилавост на том узорку најслабија.

При макро испитивању јасно се види да на узорку 3 узорци имају највише изражену порозност, као и геметријуску нетачност. У табели 8. приказано је поређење просечне висине сваког слоја и просечног уноса топлоте код узорка. Из таблице је видљиво да код узорка 1 имамо нешто већу просечну висину у односу на друга два узорка.

Такође видимо да је унос топлоте узорка 1 сличан узорку 3 иако су параметри наваривања напон, струја и брзина наваривања различити. Може се констатовати да је највећа вредност просечне висине слоја код узорка 1 у односу на друга два узорка добијена због утицаја мале јачине струје као и нижег напона лука.

То значи да када су вредности веће од оптималних долази до већег претапања предходног слоја као и цурења са стране.

Табела 8. Просечна висина слоја и унесена топлота за сва три узорка

Узорак	Просечна висина слоја, [mm]	Просечан унос топлоте, [kJ/cm]
Узорак 1	1,54	2,71
Узорак 2	1,45	3,05
Узорак 3	1,34	2,68

5. ЗАКЉУЧАК

На основу спроведених испитивања и анализа резултата може се закључити следеће:

- Током анализе резултата може се приметити да су најбољи резултати добијени на узорку 1 и 2, па се може закључити да мала јачина струје са оптималном брзином заваривања и напона омогућава WAAM технологији мањи број пролаза до задате висине, бољу распоређеност растопа, мање прскање и изливање, као и готово минималну порозност. Треба напоменути да је узорак 1 рађен у једном правцу и да је јасно видљив утицај топлоте на почетку заваривања и на крају па је и сама геометрија узорка другачија у односу на друга два узорка која су рађена са наизменичним наваривањем.

- На узорку 3 је уочена знатно већа порозност и већи број пролаза као и веће прскање у односу на остала два узорка. Сматра се да је разлог те порозности већа брзина заваривања, већа јачина струје и напона као и утицај средине.

6. ЗАХВАЛНОСТ

Резултати презентовани у овом раду су реализовани у оквиру пројекта „Истраживање и примена савремених метода карактеризације материјала и заварених спојева у производном машинству“, Департмана за производно машинство, ФТН Нови Сад.

7. ЛИТЕРАТУРА:

- [1] Палић В., Сабо Б.: Технологија заваривања, Факултет техничких наука, Нови Сад 2003
- [2] Група аутора: Поступци заваривања и опрема за заваривање (наваривање), Завод за заваривање, број док.: 1.14. стр. 1, 2004.
- [3] Стандард SRPS EN ISO 15614-1:2017

Кратка биографија:



Милан Радовановић рођен у Новом Саду 1992. године. Мастер рад на Факултету техничких наука из области Машинство – Материјали и технологије спајања одбранио је 2019. године.



Проф. др. Себастијан Балаш рођен је у Сомбору 1974. год. Докторирао је на Факултету техничких наука 2010. год., а 2016. год. је биран у звање ванредног професора из уже научне области Материјали и технологија спајања.

ЕКСПЕРИМЕНТАЛНО ОДРЕЂИВАЊЕ УТИЦАЈА ОКСИДНИХ ТОПИТЕЉА НА ТИГ ПОСТУПАК ЗАВАРИВАЊА**EXPERIMENTAL EXAMINATION OF THE INFLUENCE OF OXIDE FLUXES ON GTAW WELDING METHOD**

Драган Перановић, Себастиан Балош, Факултет техничких наука, Нови Сад

Област – МАШИНСТВО

Кратак садржај- У раду је извршено експериментално испитивање оксидних топитеља на дубину увара код ТИГ заваривања. Испитан је ефекат оксидних топитеља на бази нано и микро честица титан-диоксида при заваривању аустенитног нерђајућег челика. Показано је да су нано честице ефикасније у повећању дубине увара у односу на микрочестице, као и да је највећа дубина увара добијена код мешавине микро и нано честица. Основни механизам за повећање дубине увара је обрнута Марангонијева конвенција. При томе, материјал метала шава уместо да тече од центра метала шава према ивицама, тече од ивица према средини и према доле, што резултује смањењу ширине и повећању дубине увара.

Кључне речи: ТИГ заваривање, оксидни премаз, повећање увара

Abstract: In this paper, the experimental examination of the oxide fluxes influence on penetration depth in GTAW welding was done. The effect of oxide fluxes based on nano and micro titanium-dioxide particles on austenitic stainless-steel welding was studied. It was shown that nano particles are more effective than micro particles, as well as that the mixture of the two results in the highest penetration. The basic mechanism that influences the increase in penetration depth is the reversal of Marangoni convection. That means, the melt of the weld metal flows from the edges to the center and towards the depth of the base material, instead of from the center to the edges, resulting in a narrower weld of the higher penetration.

Keywords: GTAW welding, oxide layer, penetration increase

1. УВОД

Заваривање представља технологију помоћу које се врши спајање два или више, сродних или разнородних материјала [1]. У новије време све се више користе поступци заваривања који омогућавају да се изврши заваривање материјала у различитим, специфичним условима. Такође, у данашње време примена ТИГ (tungsten inert gas) поступка је знатно већа, највише

због врхунског квалитета споја, који се, између осталог, постиже бољом контролом унесене топлоте и додатног метала захваљујући раздвајању улога додатног метала и електроде. Заваривање ТИГ поступком је могуће и без додатног материјала, што је посебно важно код танких лимова [2]. Основни недостатак ТИГ поступка је релативно мали увар. То значи, да је за добијање заварених спојева код лимова већих дебљина потребно заваривање из више пролаза, што узрокује велику потрошњу додатног материјала и заштитних гасова. Тренутно најпогодније решење за повећање дубине увара код ТИГ поступка је примена активних топитеља.

2. ЕКСПЕРИМЕНТАЛНИ РАД

Експериментални део обухватао је заваривање плоча од нерђајућег челика уз присуство активног топитеља са различитим односима TiO_2 честица нано и микро величине. Основни материјал је нерђајући челик AISI 304 (X5CrNi18-10) дебљине 10 и ширине 50 mm. Циљ експеримента био је да се уз помоћ активног топитеља оствари промена геометрије зава на тај начин што ће завар имати већу дубину увара уз истовремено смањење ширине.

Хемијски састав основног материјала приказан је у наставку:

- <0,03 % C
- 0,4 % Si,
- 1,6 % Mn,
- <0,0006 % S,
- 17,73 % Cr,
- 0,03 % P,
- 0,011 % Al,
- 0,38 % Cu,
- 8,88 % Ni,
- 0,26 % Mo,
- 0,012 % Sn,
- 0,07 % V,
- Fe - остатак

За потребе експеримента заваривање је обављено са два уређаја. Први уређај је извор струје за заваривање *Alfain Pegas 200 AC/DC Pulse PFC* са максималном снагом уређаја од 200 A, а други уређај је трактор за

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији ментор је био др Себастиан Балош, ванр. проф.

заваривање *Lorch Track Wheel*, помоћу којег се оставарује уједначено кретање пиштоља за заваривање током процеса.

Први део експеримента који је обухватао припрему и заваривање узорака извршен је у фирми Proficut d.o.o у Бачком Петровцу док је други део који чини карактеризација завареног споја обављен у Лабораторији за испитивање материјала на Факултету техничких наука у Новом Саду.

Припрему материјала обухватало је чишћење материјала помоћу медицинског алкохола, етанола (C_2H_5OH). Након чишћења површине материјала уследило је обележавање материјала тј. укупна дужина плоче од 350 mm подељена је на седам једнаких делова од по 50 mm. На шест од седам делова наносени су активни топителџи са различитим односима микро и нано TiO_2 честица (номиналне величине 0,3 μm и 20 nm) док је последњи, седми, део плоче имао је улогу контролног узорка. Припремљено је шест различитих мешавина активног топителџа при чему свака садржи 5 масених % честица растворених у ацетону ($(CH_3)_2CO$):

1. 5 % нано честица, са ознаком (5H)
2. 4 % нано и 1 % микро честица (4H1M),
3. 3 % нано и 2 % микро честица (3H2M),
4. 2 % нано и 3 % микро честица (2H3M),
5. 1 % нано и 4 % микро честица (1H4M)
6. 5 % микро честица (5M)
7. контролни узорак на којем нема активног топителџа (0)

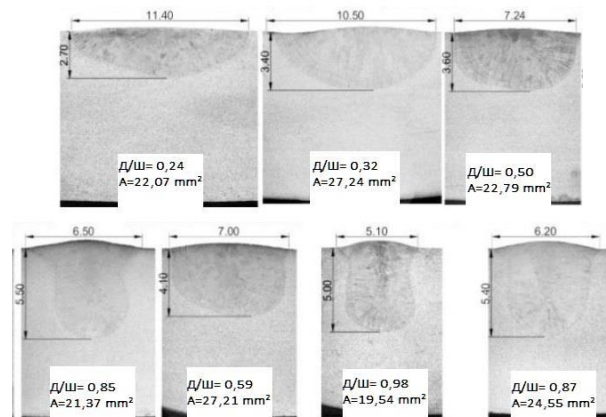
За мерење масе употребљена је аналитичка вага *Tehnica Type 2615* тачности 0,01 mg, док је за мешање честица са растварачем употребљена магнетна мешалица *Tehnica MM530*. Дужина мешања компоненти износила је десет минута. Активни топителџ је наносен на узорке ручним путем уз помоћ четкице ширине 10 mm, а након сушења активног топителџа уследило је заваривање. Заваривање је извршено дуж радног комада прелазећи при том све делове узорка на који је нанет активни топителџ као и контролни узорак. Заваривање је обављено једносмерном струјом директне поларности. Пречник млазнице на пиштољу за заваривање износио је $\varnothing 12,7$ mm. Размак између електроде и основног материјала је износио 2 mm. Процес заваривања одвијао се са нетопљивом волфрамовом електродом са 2 % оксида торијума (црвена ознака), док је пречник електроде био $\varnothing 2,4$ mm. При процесу заваривања употребљена је затупљена електрода са углом врха од 90° и затупљењем при врху пречника од 0,5 mm. Као заштитни гас употребљен је аргон са протоком од 12 l/min, док је брзина заваривања износила 100 mm/min. Процес карактеризације завареног споја обухватао је макро испитивање, испитивање микроструктуре и испитивање микро тврдоће. Најпре је извршено резање, монтирање, брушење (P240 до P2500) и полирање (дијамантским суспензијама гранулације од 6 μm до 1 μm) на лабораторијској опреми *Struers*. Завршни процес припреме узорка за испитивање обухватао је нагризање узорка царском водом ($HNO_3 + 3HCl$). Затим је на узорцима измерена

ширина и дубина увара на основу којих је израчунат однос између дубине и ширине завареног споја. Такође, изглед микроструктуре је праћен на карактеристичним местима као што је метал шав, зона утицаја топлоте (ЗУТ) и основни материјал. Ове карактеристичне зоне испитане су уз помоћ светлосног микроскопа *Leitz Orthoplan*, који је употребљен и за прецизно мерење дубине пенетрације и ширине завара. За мерење микротврдоће по Викерс методи употребљен је уређај *WilsonTukon 1102*. Мерење микротврдоће по Викерсу је обухватало мерење на 1 mm испод површине узорка (линија 1), 1 mm од максималне дубине увара (линија 2) и једна линија мерење по оси завара (линија 3) а сила оптерећења износила је 9,8 N.

3. РЕЗУЛТАТИ

3.1. Макро испитивање

Примери макро испитивања за узорке код којих је примењен активни топителџ као и контролни узорак приказани су на слици 1. Поред тога означене су и измерене вредности дубина пенетрације, ширина завара, однос дубине и ширине завара и површина завара. Код контролног узорка (0) готово нема надвишења завара, дубина пенетрације је најмања а однос дубине према ширини завара је најмањи. При употреби активног топителџа који садржи само оксидне честице микронске величине (5M) и топителџа са оксидним честицама претежно микронске величине (1H4M) добија се елиптични облик завара уз знатну промену од односа дубине према ширини завара у односу на контролни узорак. Код узорка који садрже уравнотежен садржај оксидних честица микро и нано величине и узорка код којих је примењен топителџ са претежно нано честицама добија се израженија промена геометрије завара. Почевши од најмање израженог, скоро полукружног облика код узорка 5M до узорка 4H1M код којег се добија највећа вредност дубине завара и најмања ширина уз најмањи однос између дубине и ширине завара. Код узорак 3H3M, 2H3M и 4H1M примећује се да се са малом променом садржаја микро и нано оксидних честица добија велика промена у геометрији завара.



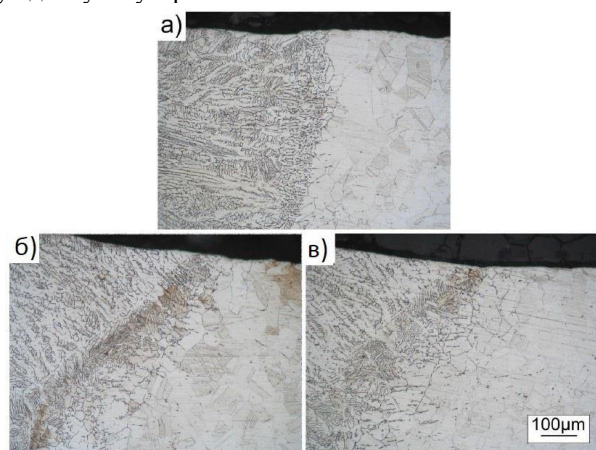
Слика 1. Приказ макро испитивања узорка

Активни топителџ који се састоји само од оксидних честица нано величине (5H) даје мању дубину увара

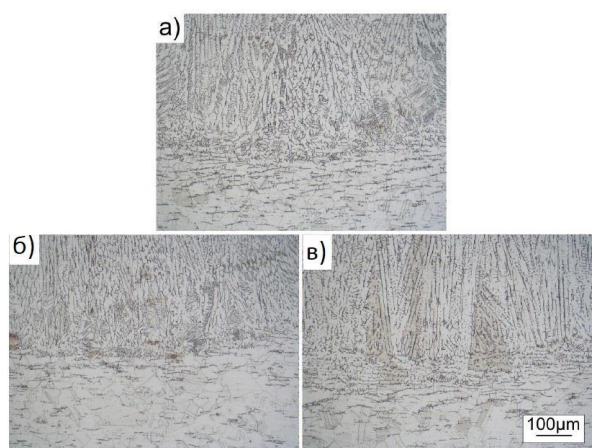
једино у односу на топитељ (2Н3М), док у односу на остале има већу дубину увара. Што се тиче односа дубине и ширине завава вредност је приближно 0,9 и по том параметру облик завава је доста сличан узорцима (2Н3М, 4Н1М).

3.2 Микроструктура

Феномен који се односи на микроструктурне карактеристике завареног споја, приказан је на неколико репрезентативних узорача као што су 0, 5М и 5Н. Узорци 5М и 5Н су узорци код којих је примењен активни топитељ. На основу резултата односа дубине и ширине код узорка 1М4Н долази се до закључка да је овај узорак највише склон појави локалне сегрегације у средњем делу завава. На слици 2. приказана је микроструктура узорача 0, 1М4Н и 5Н. На основу резултата може да се констатује да све микроструктуре одговарају типичној дендритској морфологији која се иначе налази код заварених спојева нерђајућег челика. Иако, код свих узорача дендрити изгледају слично издвајају се 1М4Н и 5Н код којих је дендритска структура нешто финија у односу на узорак 0.



Слика 2. Микроструктура завареног споја на површини линије стапања: а) Узорак 0; б) Узорак 1М4Н; в) Узорак 5Н



Слика 3. Микроструктура завареног споја на дну завава: а) Узорак 0; б) Узорак 1М4Н; в) Узорак 5Н

Микроструктура у близини линије стапања приказана је на сликама 2. и 3. На слици 2. приказана је микроструктура у близини линије стапања на површинском делу завава, док је на слици 3.

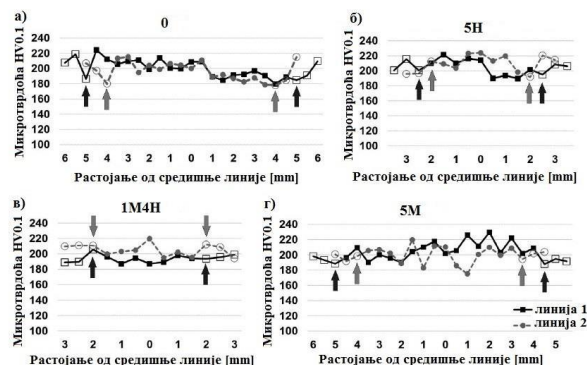
приказана линија стапања на дну завава. На основу резултата може да се примети да постоји значајна разлика између узорача добијених без присуства активног топитеља 0 и узорача код којих је употребљен активни топитељ при заваривању 1М4Н и 5Н. Код узорка 0 примењена је на површини завава појава крупнијих аустенитних зрна, док се та појава код узорача код којих је примењен активни топитељ 1М4Н и 5Н испољава на дну завава, испод линије стапања у основном материјалу.

3.3. Микротврдоћа

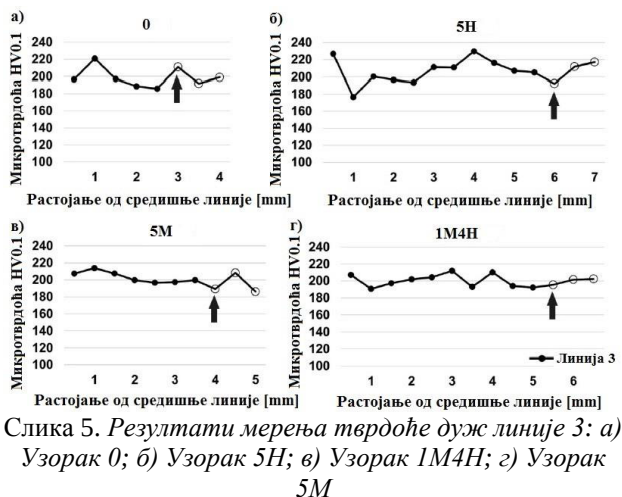
Испитивање микротврдоће обављено је на узорцима на којима је вршено испитивање и микроструктуре. Добијене вредности микротврдоће приказане су на сликама 4. и 5. Резултати који се налазе на слици 5. представљају вредности микротврдоће добијене мерењем дуж линије 1 и 2. Вредност микротврдоће измерене на линији 3 приказане су на слици 5. Минималне вредности микротврдоће за узорак (0) добијају се у близини површине стапања материјала, испод површине узорка (линија 1). Код узорача 5М, 1М4Н и 5Н минималне вредности добијају се на мерној линији 3.

4. ДИСКУСИЈА

Активни топитељи са различитим саставом нано и микро честица нису имали утицај на површину завава. Ово је у супротности са резултатима које су приказали Тсенг и Лин [33], који су успели да значајно повећају површину завареног споја уз примену активног топитеља на бази SiO_2 и Al_2O_3 наночестица у поређењу са употребом честица микро величине. Највећа дубина пенетрације постигнута је код узорка 3М2Н, а приближна вредност остварена је код узорка 5Н. Овај резултат делимично се поклапа са резултатима који су добијени у раду [33]. Наиме, у раду [33] употреба SiO_2 наночестица је имала ефикаснији утицај у односу на честице микро величине, док се при примени Al_2O_3 наночестица није приметилa значајна разлика у односу на честице микро величине. У овом раду употреба TiO_2 наночестица показала се ефикаснијом у односу на микро честице. Међутим, најбољи резултат добијен је са мешавином нано и микро оксидних честица.



Слика 4. Резултати мерења тврдоће дуж линија 1 и 2: а) Узорак 0; б) Узорак 5Н; в) Узорак 1М4Н; г) Узорак 5М



Сличан тренд примећен је и при праћењу односа дубине и ширине шав. Модел течења материјала при заваривању ТИГ односно А-ТИГ поступком приказан је на слици 6.

Код узорака добијених заваривањем ТИГ поступком тврдоћа у близини линије стапања материјала и близини површине завара има значајно мање вредност у поређењу са узорцима код којих је употребљен активни топител.

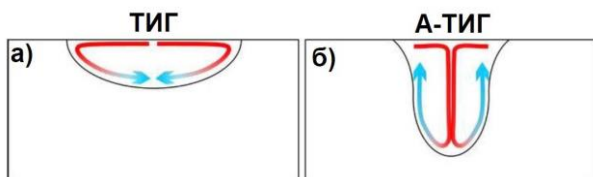
Ова констатација се подудара и са добијеном микроструктуром у овим зонама. Наиме, крупна аустенитна зрна која се јављају у самом завару одговарају мањим вредностима измерене тврдоће у овој области.

Све ово указује на то да растопљени метал који настаја конвенционалним ТИГ заваривањем има кретање према линији стапања односно према основном материјалу преносећи при том топлоту на основни материјал.

Ова топлота узрокује раст аустенитних зрна. На овај начин долази до повећања аустенитних зрна само на линији стапања док у зони испод завара аустенитна зрна остају непромењена.

Обрнута Марангонијева конвекција код узорака код којих је примењен активни топител узрокују да растопљени метал има кретање ка унутра и на тај начин продире у основни материјал. Ово узрокује знатно већу дубину пенетрације а као спољашњи ефекат јавља се пренос топлоте на основни материјал и као резултат добијају се крупна аустенитна зрна у области испод завара.

Насупрот томе, аустенитна зрна у близини линије стапања са основним материјалом остају непромењена јер је растопљен метал који је достигао ово подручје већ пренео део своје топлоте.



Слика 6. Модел течења метала код ТИГ односно А-ТИГ поступка: а) Без употребе активног топител; б) Употреба активног топител

5. ЗАКЉУЧЦИ

Према резултатима представљеним у оквиру овог рада, могу да се донесу следећи закључци.

- Наночестице су се показале ефикасније за повећање дубине пенетрације у односу на микрочестице док се најефикасније показала мешавина нано и микрочестица.

-Код узорака код којих није примењен активни топител дошло је до укрупњавања величине зрна у близини линије стапања испод површине основног материјала, резултирајући на тај начин смањеном тврдоћом у овој зони.

-Код узорака који су заваривани уз присуство активног топител такође је дошло до укрупњавања величине зрна али у зони исподповршине завара, резултирајући на исти начин смањење тврдоће у овој зони.

-Главни узрок смањења тврдоће и повећања величине зрна испод површине основног материјала и испод површине завара може да се припише течењу материјала високе температуре у близини ове области услед чега долази до преноса топлоте на основни материјал.

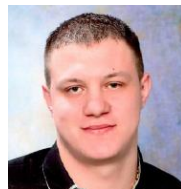
6. ЗАХВАЛНОСТ

Резултати презентовани у овом раду су реализовани у оквиру пројекта „Савремене методе испитивања у науци о материјалима и заваривању у производном машинству“, Департмана за производно машинство, ФТН Нови Сад.

7. ЛИТЕРАТУРА:

- [1] Митровић М.: Наваривање тврдих легура, Факултет техничких наука, Нови Сад, 2015.
- [2] Тордаш Д.: Орбитално заваривање нерђајућих челика, Пула, Пула, 2007.
- [3] К.-Н. Tseng, С.-Y. Hsu, Performance of activated TIG process in austenitic stainless steel welds, J. Mater. Process. Tech. 211 (2011) 503–512.

Кратка биографија:



Драган Перановић рођен у Теслићу 1994. године. Мастер рад на Факултету техничких наука из области Машинство – Материјали и технологије спајања одбранио је 2019. године.



Проф. др Себастијан Балог рођен је у Сомбору 1974. год. Докторирао је на Факултету техничких наука 2010. год., а 2016. год. је биран у звање ванредног професора из уже области Материјали и технологија спајања.

RAZVOJ IDEJNOG REŠENJA I KONSTRUKCIJE ALATA ZA IZRADU FITINGA LUKOVA Ø110/15° Ø110/30°

DEVELOPMENT AND DESIGN OF MOLD FOR INJECTION MOLDING OF FITTINGS Ø110/15° Ø110/30°

Zvonko Magoč, Mladomir Milutinović; *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj – U ovom radu prikazano je konstrukciono rešenje alata/kalupa za injekciono presovanje fittinga lukova Ø110/15° i Ø110/30°, koji predstavljaju standardne elemente kanalizacione instalacije. Pored toga, primenom softverskog paketa Moldex3D izvršena je analiza procesa po fazama u cilju provere validnosti konstrukcije alata i usvojenih tehnoloških parametara.

Abstract – In this paper a design solution for mold for injection molding of Ø110/15° and Ø110/30° fittings is given. Those fittings are standard elements of a sewer system. In addition, the mold design and the molding process parameters were analyzed using software package.

Ključne reči: Injekciono presovanje, luk, numerička simulacija, Moldex3D.

Key words: Injection molding, fitting numerical simulation, Moldex 3D.

1. UVOD

Polimeri su materijali koji se sastoje iz većeg broja molekula – monomera. Polimer je složenica grčkog porekla sastavljena od reči poli – mnogo i meros – deo. Polimeri se sastoje iz većeg broja monomera (mono – jedan) grupisanih u složeniji lančasti molekul i međusobno su polarizovani (povezani u veće agregate)

Sintetički materijali, koji imaju amorfnu makromolekularnu strukturu, čine veliku grupu konstrukcionih materijala koji se jednom rečju nazivaju plastike. Za proizvodnju plastičnih masa koriste se jedinjenja organskog i mineralnog porekla koja predstavljaju polazne sirovine. Organske sirovine mogu biti biljnog ili životinjskog porekla. Mineralne sirovine se koriste kao polazni materijal za dobijanje hemijskog jedinjenja koja se različitim procesima prevode u poluproizvode u obliku granula, tableta, praha, smole, tečnosti [1]. Sirovine mineralnog porekla su [1]:

1. Nafta,
2. Ugalj i
3. Zemni gas.

NAPOMENA:

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

Analiza plastičnih masa izvodi se praćenjem opštih karakteristika, načina oblikovanja, dodatnih materijala kao i njihov zadatak, a najznačajnija klasifikacija plastičnih masa je prema glavnim hemijskim familijama i prema njihovoj prirodi.

Sve plastične mase, uzimajući za osnovu smolu, dele se na više osnovnih grupa prema tabeli (tabela 1.).

Tabela 1. Pregled osnovnih plastičnih masa [1]

Vrsta plastične mase	Familija – hem. priroda	Vrsta plastične mase	Familija – hem. priroda
Termo- plastične plastične mase	Polietileni	Termo- reaktivne plastične mase	Fenoli
	Polipropileni		Aminoplasti
	Polivinilhlorid		Epoksidne smole
	Polistirol		Poliestri*
	Poliamidi		Urelici
	Poliformaldehid		
	Polikarbonati		
	Celuloze		
	Metakrili		
	Politetrafluoretileni		
	Acetati		

Napomena *Pripadaju i jednoj i drugoj grupi plastičnih masa [2].

2. PRERADA PLASTIČNIH MASA POSTUPKOM INJEKCIJNOG PRESOVANJA

Injekciono presovanje se zasniva na omekšavanju pripremljene plastične mase u zagrejanom cilindru i njenim periodičnim ubrizgavanjem u temperiranu kalupnu formu u kojoj dolazi do otvrdnjavanja proizvoda (slika 1.).

Proces injekcionog presovanja sastoji se od više faza. Prva faza ciklusa je zatvaranje kalupa, a ostvaruje se jedinicom za zatvaranje kalupa. Za tu operaciju potrebno je određeno vreme zatvaranja kalupa.

Podesivi profili brzine i sile zatvaranja služe kako bi se alat zaštitio od oštećenja.

U slučaju brizganja komada sa šupljinom, jezgra se takođe uvlače nakon zatvaranja alata, a mašina uspostavlja veliku silu zatvaranja.

Sledi približavanje mlaznice ka dizni alata, mlaznica jedinice za brizganje prislonjena je na alat, a zatim se podiže pritisak kontakta kako bi sprečio povratak ulivene mase.

U tom trenutku element za ubrizgavanje, u ovom slučaju pužni vijak, deluje kao klip. Pritisak ubrizgavanja može biti između 300 i 2500 bara u zavisnosti od alata i vrste plastomera. Ubrizgavanjem počinje i hlađenje rastopa u kalupnoj šupljini.

Završetkom punjenja kalupne šupljine dolazi do prelaska sa pritiska ubrizgavanja na naknadni pritisak u cilindru za ubrizgavanje. Naknadni pritisak je po pravilu niži od pritiska ubrizgavanja, jer više ne postoji visok otpor tečenja rastopa. Njegov zadatak jeste da izravna (ispegla) komad dopunskim punjenjem kalupne šupljine.

Naknadni pritisak deluje do "pečaćenja" tj. do trenutka kada se spojno mesto između kalupne šupljine i ulivnog sistema toliko ohladi i očvrstne da više nije moguće proticanje rastopa.

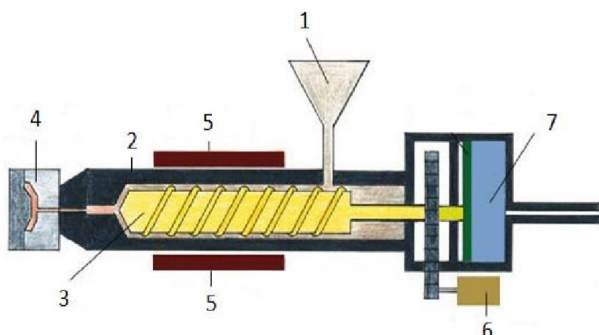
Time je omogućeno odmicanje mlaznice od ulivnog kanala i započinjanje plastifikacije.

Uporedno sa plastifikacijom, a pri odmaknutoj mlaznici, dolazi do uzimanja potrebne količine plastomera za naredni ciklus. Plastomer u obliku granulata se pretvara u rastop prolazeći duž rotirajućeg pužnog vijka.

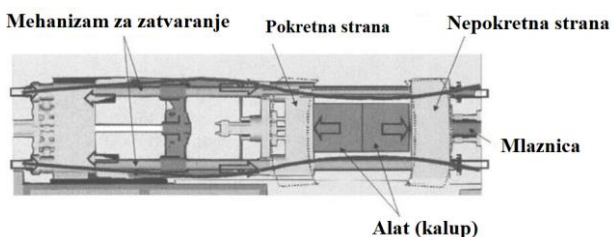
Toplota potrebna za zagrevanje plastomera dovodi se grejačima smeštenim na cilindru za topljenje. Potrebno vreme hlađenja otpreska često je duže od vremena ubrizgavanja, naknadnog pritiska, plastificiranja i vraćanja mlaznice. Zbog toga je potrebno dopunsko vreme hlađenja da se otpresak ohladi na temperaturu koja omogućuje pouzdano potiskivanje otpreska iz kalupne šupljine.

Kada se otpresak dovoljno ohladi, kalup se otvori odmicanjem pokretnog od nepokretnog dela. Tokom te faze se potiskuje otpresak iz kalupne šupljine.

Ciklus se završava istekom vremena za pripremu za naredni ciklus, nakon čega je mašina spremna za novi ciklus. Na narednoj slici (slika 2.) prikazana je šema mehanizma za zatvaranje alata [3].



Slika 1. Šema mašine za brizganje plastike: 1 - Koš, 2 - Cilindar, 3 - Puž, 4 - Kalup, 5 - Grejači, 6 - Motor, 7 - Hidraulični fluid [1]



Slika 2. Šema mehanizma za zatvaranje alata [3]

3. ALATI ZA INJEKCIONO PRESOVANJE

Alati za preradu plastičnih masa su tipičan primer kompleksnih alata. Od njih se očekuje da budu efikasni i pouzdani u njihovoj eksploataciji. Izrada kalupa zahteva određeno vreme i predstavlja veoma bitnu fazu koja se sastoji iz nekoliko podfaza. Kalup je najpre potrebno dizajnirati, zatim izraditi njegove delove, sklopiti u jednu funkcionalnu celinu i na kraju izvršiti njegovu probu. Sa tog aspekta kalupi za preradu plastičnih masa treba uspešno da odgovore na nekoliko zadataka i zahteva [4]:

- Tehnološki – u jednom radnom ciklusu kompletno oblikovati jedan ili više proizvoda, preuzeti rastopljenu masu, razdeliti je, popuniti kalupne šupljine, ohladiti je, prevesti je u čvrsto stanje i izbaciti proizvod,
- Konstrukcijski – treba preuzeti sile bez deformacija, osigurati potrebna kretanja alata pri otvaranju/zatvaranju i izbacivanju proizvoda kao i osigurati tačno vođenje delova alata i
- Funkcionalni – ulivni sistem, sistem oblikovanja i odzračivanja, sistem za temperiranje (hlađenje, grejanje), izbacivački sistem, vođenje i centriranje, prihvatanje prese, prihvatanje opterećenja i prenos potrebnog kretanja [4].

4. S LINE PROGRAM FITINGA

Komponente S line programa kanalizacione instalacije kompanije „Peštan“-Arandelovac su posebna (unapređena) linija proizvoda koje karakteriše visok kvalitet i prepoznatljivi dizajn. Proizvode se u Light Blue RAL 5012 boji, na sebi imaju "Peštan kućicu" logo firme, ali koja istovremeno ima i značajnu funkcionalnu ulogu. Naime, logo se nalazi na površini komada i definiše uglove pod kojima se postavljaju određeni fitinzi, kao što su lukovi $\varnothing 110/15^\circ$ i $\varnothing 110/30^\circ$, prikazani na slici 3.



Slika 3. Dizajn S line fittinga

U tabela 2 navedene su mehaničke karakteristike komponenti S line-a programa [5].

Osnovni sastav S line smeše:

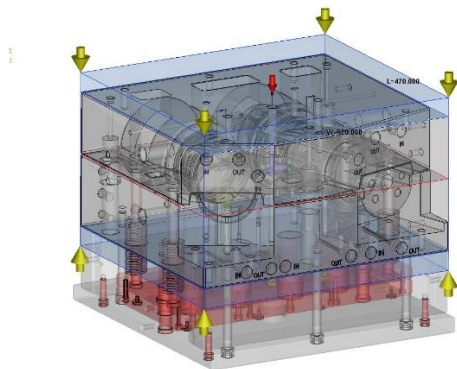
1. Borealis PP BEC 5012 – Borealis,
2. Kreda Peštan S5,
3. Listab CA – Chempson PolymerAdditive AG i
4. Masterbač - hemikalija.

Tabela 2. Mehanička svojstva S line-a fittinga [5]

Materijal	PP-H (Polipropilen kopolimer)
Gustina	Cevi (Ø32 – Ø160) – $1,3 \text{ g/cm}^3$, fitting – $1,4 \text{ g/cm}^3$
Temperaturna otpornost	Kratkotrajna do 95°C , dugotrajna do 60°C
Koeficijent linearnog širenja	$0,05 \text{ mm/m}^\circ\text{C}$
Hemijska otpornost	$\text{pH } 2 - \text{pH } 12$
Modul elastičnosti	$2400 - 3100 \text{ MPa}$
Način spajanja	Muf i gumica – otpornost na curenje do pritiska od $0,5 \text{ bar}$

5. IDEJNO REŠENJE ALATA

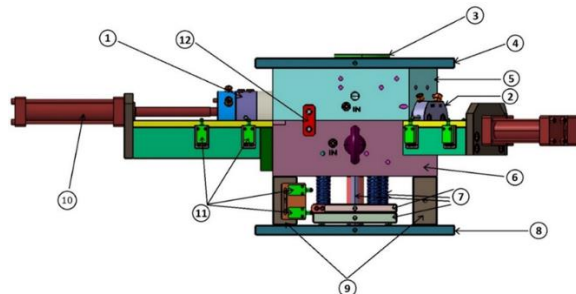
Na slici 4 dato je idejno rešenje unapređenog alata za injeksiono presovanje fittinga prikazanih na slici 3. Alat se sastoji iz dve kalupne šupljine, jedna za luk Ø110/15°, a druga za luk Ø110/30°. Podeona ravan se nalazi na sredini komada. Zbog toga što alat čine dve kalupne šupljine (gnezda), obe su opremljene sa po dva izbijača. Da ne bi ulivak ostao u alatu dodat je poseban izbijač za vađenje ulivka. U alatu se koriste jednolični okrugli izbijači. Razlog zašto se koriste dva različita gnezda je najviše ekonomski. Proizvodi su relativno novi na tržištu i njihova potražnja još nije toliko zastupljena, pa se komadi proizvode u malim serijama, svega nekoliko hiljada. Razvoj dva različita alata za svaki komad bi bio ekonomski neisplativ u ovom trenutku jer bi, s obzirom na plasman proizvoda, vreme povrata investicije (troškovi razvoja i izrade alata) bilo veoma dugo. Alat koristi hladni ulivni sistem, sa centralnim ulivkom.



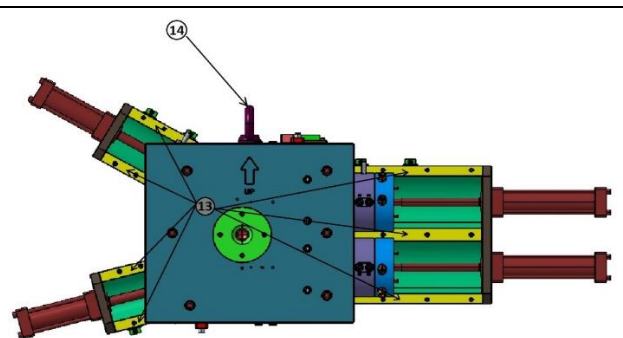
Slika 4. Idejno rešenje alata za S line fitting luk Ø110/15° Ø110/30°

6. RAZVOJ KONSTRUKCIJE ALATA S LINE FITINGA LUK Ø110/15° Ø110/30°

Razvijena konstrukcija alata data u dve projekcije, sa označenim glavnim elementima prikazana je na slikama 6 i 7.



Slika 5. Sklop alata, pogled sa strane



Slika 6. Sklop alata, pogled od gore

1. Sklop skupljajućeg trna,
2. Sklop bočnog trna,
3. Centrač,
4. Nepokretna vezivna (stezna) ploča,
5. Nepokretna gravurna (kalupna) ploča,
6. Pokretna gravurna (kalupna) ploča,
7. Izbijački sistem (noseća izbijačka ploča – gornja i potporna izbijačka ploča – donja,
8. Pokretna vezivna (stezna) ploča,
9. Gređice (odstojnici),
10. Hidraulični cilindri,
11. Mikroprekidači,
12. Mehanička blokada (blokada),
13. Klizne površine bočnih i skupljajućih trnova, na nosačima trnova i
14. Kuka za transport.

7. SIMULACIJA PROCESA INJEKCIONOG PRESOVANJA S LINE FITINGA LUK Ø110/15° Ø110/30°

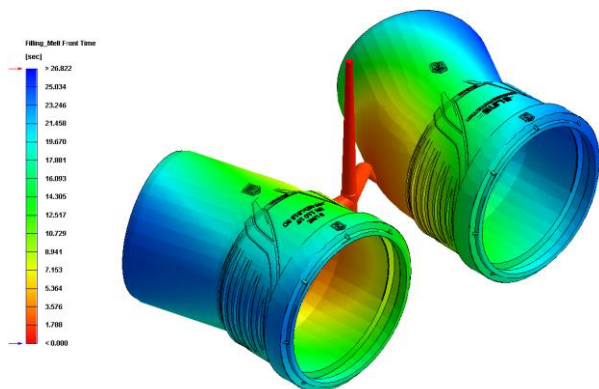
Simulacija procesa injeksionog presovanja je urađena u programskom paketu Moldex 3D.

U programskom paketu Moldex 3D proces se analizira kroz nekoliko faza ciklusa:

1. Predgrevanje alata tj. kalupnih (gravurnih) ploča,
2. Punjenje kalupnih šupljina,

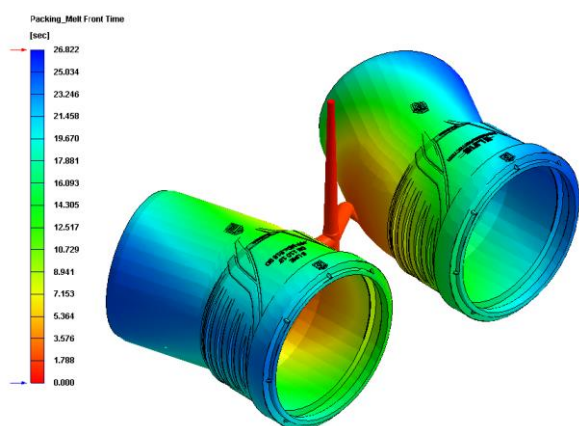
3. Delovanje naknadnog pritiska,
4. Hlađenje,
5. Deformacije otpreska u kalupu.

Sa slike (slika 7.) može se videti da je vreme popunjavanja kalupne šupljine dobijeno na osnovu simulacije 26,8 sekundi, da je popunjavanje 100%, tj. da je kalup popunjen u celosti, što predstavlja identičan rezultat kao u praksi. Vreme brizganja teži da se smanji što je više moguće. Kada su manji pritisci i brzina popunjavanja, duže traje popunjavanje kalupne šupljine i može doći do pojave fleka. Uzrok tome jeste prisustvo krede u smeši.



Slika 7. Vreme popunjavanja kalupne šupljine

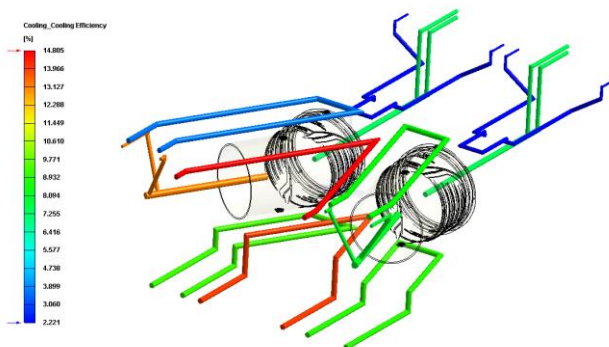
Naknadni pritisak (slika 8.) je produženo delovanje pritiska u cilindru radi popunjavanja šupljina nastalih skupljanjem materijala pri hlađenju. Vrednost naknadnog pritiska treba da bude od 60% do 70% od pritiska brizganja. Vreme trajanja naknadnog pritiska u praksi je do 2 sekunde za prikazane komade. Generalno, vreme trajanja naknadnog pritiska zavisi od debljine zida. Prema standardu EN 1451 za debljinu zida 1,8 mm potrebno je vreme 1,8 - 2 sekunde naknadnog pritiska. Pri hlađenju delova istovremeno dolazi do njihovog skupljanja. Usled nedovoljne vrednosti naknadnog pritiska može doći i do formiranja šupljina, tj. do pojave vazduha unutar ulivka, što uzrokuje da se udubljenja pojave i na samoj površini komada.



Slika 8. Naknadni pritisak

U nastavku (slika 9.) dat je stepen iskorišćenja kanala za hlađenje. Rezultati simulacije pokazuju da je srednja vrednost stepena iskorišćenja 7,28%. Razlog zašto je

stepen iskorišćenja toliko nizak je zbog modifikovanja kanala za potrebe ispravnosti simulacije o kojoj je bilo reči na početku poglavlja. Stepenn iskorišćenja kanala za hlađenje je znatno veći u praksi.



Slika 9. Stepenn iskorišćenja kanala za hlađenje

8. ZAKLJUČAK

U ovom radu prikazan je proces injektionog presovanja S line fittinga luk $\varnothing 110/15^\circ$ i luk $\varnothing 110/30^\circ$ zajedno sa razvijenom konstrukcijom alata za njihovu izradu postupkom injektionog presovanja. Na osnovu 3D modela fittinga u programskom paketu Moldex 3D numeričkom simulacijom sprovedena je analiza prema stvarnim parametrima iz proizvodnje.

Rezultati dobijeni numeričkom simulacijom se u velikoj meri poklapaju sa rezultatima realnog procesa što pokazuje da su numeričke simulacije pouzdane inženjerske alatke i da se u tom smislu mogu koristiti u budućnosti kako za projektovanje tehnološkog procesa, tako i za projektovanje samih alata.

9. LITERATURA

- [1] Milorad Jovanović, Vukić Lazić, Dragan Adamović, Neda Ratković: Mašinski materijali, Mašinski Fakultet, Kragujevac, 2003.
- [2] Bogdan Nedić, Vladislav Đukić: Plastične mase, Mašinski Fakultet, Kragujevac, 2004.
- [3] Bogdan Nedić, Nataša Vesić, Darko Vasiljević: Boja, kolometrija i plastične mase, Mašinski Fakultet, Kragujevac, 2018.
- [4] Boško Perošević: Kalupi za injektiono presovanje plastomera (termoplasta), naučna knjiga, Beograd, 1995.
- [5] www.Pestan.net/wp-content/uploads/2018/08/2018-PIPING_SOLUTIONS-SRB-web.pdf (13.05.2019.)

Kratka biografija:



Zvonko Magoč rođen u Vrbasu.

Diplomirao 2017. godine na Fakultetu tehničkih nauka, smer mašinstvo. Zaposlen u kompaniji „Peštan” na poziciji WCM lidera.

Mladimir Milutinović rođen je u Arilju 1967. godine Doktorirao je na Fakultetu tehničkih nauka 2013. god., iste godine izabran je u zvanje docenta. Oblast interesovanja su tehnologije plastičnog deformisanja, aditivne i virtuelne tehnologije

POLIMERNI MATERIJALI U MEDICINI I FARMACIJI I POSTUPCI NJIHOVE PRERADE**POLYMER MATERIALS IN MEDICINE AND PHARMACY AND PROCEDURES OF THEIR PROCESSING**

Vanja Vidović, Mladomir Milutinović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj – Polimerni materijali jedni su od najvažnijih materijala koji se koriste u medicini i farmaciji. U radu je dat prikaz materijala koji se najčešće upotrebljavaju, njihove osobine kao i primeri njihove primene u praksi. Takođe, opisani su postupci prerade ovih materijala.

Ključne reči: polimeri, biomaterijali, postupci prerade, injekciono presovanje, ekstrudiranje, kalandriranje.

Abstract – Polymer materials are one of the most important materials used in medicine and pharmacy. The paper presents the most frequently used materials, their properties and application in practice. Also, procedures for processing these materials are described.

Keywords: polymers, biomaterials, processing methods, injection molding, extrusion, calendaring

1. UVOD

Druga polovina prošlog i početak ovog veka obeleženi su intenzivnim razvojem tehnologije proizvodnje baznih plastičnih (polimernih) materijala i prerade u elemente različitog upotrebnog karaktera. Nove formulacije poznatih hemijskih jedinjenja i novih struktura aditiva, jednostavan i brz postupak oblikovanja, stvorili su ogroman prostor za korišćenje elemenata od polimera u svim segmentima industrije i u svakodnevnom životu. Najveći broj predmeta koji okružuje svakog čoveka u bilo kojoj zemlji i na bilo kojem kontinentu izrađen je, delom ili celosti, od nekog polimernog materijala. Zamenjuje drvo, aluminijum, čelik, staklo, tekstil, kožu i druge prirodne materijale. [4]

Tako masovno područje primene polimernih materijala stvorilo je osnovu za neverovatno brz razvoj raznih tehnologija prerade polimernih materijala i mašina koje je prate, a u okviru toga, razvoj konstrukcija i materijala od kojih se proizvode alati, hidraulike koja ih pogoni, robota koji opslužuju mašine i elektronike koja upravlja celokupnim tehnološkim postupkom. Takođe, polimeri se smatraju važnim medicinskim i farmaceutskim materijalom iz određenih razloga kao što su: kompatibilnost i sposobnost da izdrži dugotrajno izlaganje ljudskom telu, biodegradabilnost, mala težina, lakoća izrade i upotreba. Detaljan opis materijala kao i njihova primena prikazana je u nastavku teksta.

NAPOMENA:

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

2. POLIMERNI MATERIJALI I NJIHOVA PRIMENA

Polimeri su organska ili neorganska jedinjenja izgrađena od molekula velikih molekulskih masa (makromolekula) u kojima su atomske grupe, tzv. "strukturne jedinice", povezane hemijskim vezama čineći tako polimerni (makromolekulski) lanac. Prilikom odabira polimernog materijala treba voditi računa o strukturi polimera, hemijskom sastavu i fizičkom stanju jer oni utiču na svojstva polimera. Neka od svojstava i zahteva polimernih farmaceutskih i medicinskih materijala su [3]:

- čisti materijali
- lako preradljivi
- dugoročno raspoloživi
- imaju raspoložive sertifikate zdravstvenih organizacija

Materijali koji se najčešće upotrebljavaju u medicini i farmaciji su: polietilen (PE), polietilen-tereftalat (PET), polipropilen (PP), polivinilhlorid (PVC), poli-etar-etar-eton (PEEK) koji su u detaljno opisani u nastavku teksta. Postoje tri osnovna tipa polietilena: polietilen niske gustine (PE-LD), linearni polietilen niske gustine (PE-LLD) i polietilen visoke gustine (PE-HD). Polietilen je postojan na veliki broj hemikalija, osim oksidirajućih kiselina, halogenih elemenata i malog broja ketona. Pri sobnoj temperaturi netopljiv je i bubri samo nakon dužeg izlaganja. PE-LD ima dobra svojstva zatezne čvrstoće, savojne žilavosti, a zadržava svoju čvrstoću i do -60°C. Kad je izložen ultraljubičastom zračenju, polietilen je podložan procesima hemijske ili toplotne razgradnje. U farmaceutskoj industriji PE-HD se primenjuje za izradu bočica u kojima se nalaze lekovi u čvrstom stanju kao što su npr. tablete, kapsule i sl. Bočice mogu biti različitih dimenzija, oblika i boja. Na slici 2.1 prikazana je bočica cilindričnog oblika.

Polietilen-tereftalat je kondenzacijski polimer, što znači da proces polimerizacije uključuje uklanjanje vode. Žilav je i izvrsne je prozirnosti i sjaja. Lagan je i postojan na napukline zbog naprežanja (tenzokoroziju). Izvrsne je dimenzijske stabilnosti i posjeduje vrlo dobro svojstvo barijernosti na vodenu paru, kiseonik i ugljendioksid. PET se može ojačati staklenim vlaknima da bi se proizvodili injekcijski proizvodi s visokom toplotnom postojanošću i visokom žilavošću. PET bočice nalaze veliku primenu pri izradi bočica za oralnu higijenu. Osim za kapi, u PET bocama se mogu čuvati lekovi u krutom stanju. Tako na primer od PET-a izrađuju se bočice u kojima se čuvaju šumeće tablete. Na slici 2.2 prikazana je PET bočica namenjena čuvanju lekova u krutom stanju.



Slika 2.1 PE-HD bočice cilindričnog oblika[2]



Slika 2.2. PET bočica za lekove[11]

Polipropilen ima mnoga slična svojstva sa polietilenom niske gustine. No, polipropilen ima višu tvrdoću i zateznu čvrstoću, elastičniji je, prozirniji i sjajniji, uz podjednaku propusnost za gasove i pare. Nedostatak polipropilena je niska žilavost, posebno pri nižim temperaturama (već pri 0 °C postaje krt). PP se upotrebljava u razne svrhe u medicinskoj industriji. Kao materijal za proizvodnju medicinske ambalaže, upotrebljava se za proizvodnju medicinskih šprica, vrećica, jednokratnih komponenti za bolnice, epruveta, čaša i itd. Na slici 2.3 prikazana je šprica od polipropilena.



Slika 2.3 Šprica od polipropilena[6]

Zahvaljujući dobroj kompatibilnosti i širokom području primene PVC je materijal koji se najčešće upotrebljava. U farmaceutskoj industriji služi za izradu PVC filmova za (blister) ambalažu, te za izradu bočica različitih oblika i veličine. Na slici 2.4 prikazana je ambalaža s PVC filmom. PVC filmovi moraju biti dovoljno čvrsti i kruti kako bi predstavljali fizičku zaštitu za farmaceutski proizvod. S druge pak strane, istiskivanje proizvoda iz blister ambalaže ne sme biti preteško. Upravo zbog tih razloga debljina PVC filmova iznosi između 0,2 i 0,3 mm u zavisnosti od veličine i oblika mjehura. [2].

Pored gore navedenih polimera, veliku primenu u izradi medicinske i farmaceutske opreme pronašli su i manje poznati materijali koji pripadaju familiji poli-aril-etar-ketona, skraćeno PAEK. [9] Iz ove grupe najpoznatiji materijali su PEEK i PEKK. Poslednjih godina, poli-etar-

etar-keton (PEEK) je postao sinonim za polimere najviše klase. To je materijal koji se koristi na visokim temperaturama (250°C) i pokazuje izvanrednu mehaničku čvrstoću, žilavost, savijanje pri čemu može podneti pritisak do 3,6 GPa. Njegove karakteristike upotpunjuje biokompatibilnost, odlična hemijska stabilnost, veoma dobra dielektrična svojstva i otpornost na gama i rentgensko zračenje. [8]



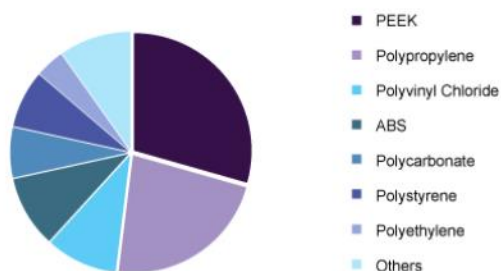
Slika 2.4 Blister od PVC-a

Pored gore navedenih polimera, veliku primenu u izradi medicinske i farmaceutske opreme pronašli su i manje poznati materijali koji pripadaju familiji poli-aril-etar-ketona, skraćeno PAEK. [9] Iz ove grupe najpoznatiji materijali su PEEK i PEKK. Poslednjih godina, poli-etar-etar-keton (PEEK) je postao sinonim za polimere najviše klase. To je materijal koji se koristi na visokim temperaturama (250°C) i pokazuje izvanrednu mehaničku čvrstoću, žilavost, savijanje pri čemu može podneti pritisak do 3,6 GPa. Njegove karakteristike upotpunjuje biokompatibilnost, odlična hemijska stabilnost, veoma dobra dielektrična svojstva i otpornost na gama i rentgensko zračenje. [8]



Slika 2.5 Proteza za zube [7]

Naravno, ubrzanim razvojem ove grane industrije dolazi do upotrebe sve većeg broja različitih polimera, od kojih su mnogi još u fazi ispitivanja. Na dijagramu je prikazana upotreba polimera u medicini i farmaciji iz 2016. godine.

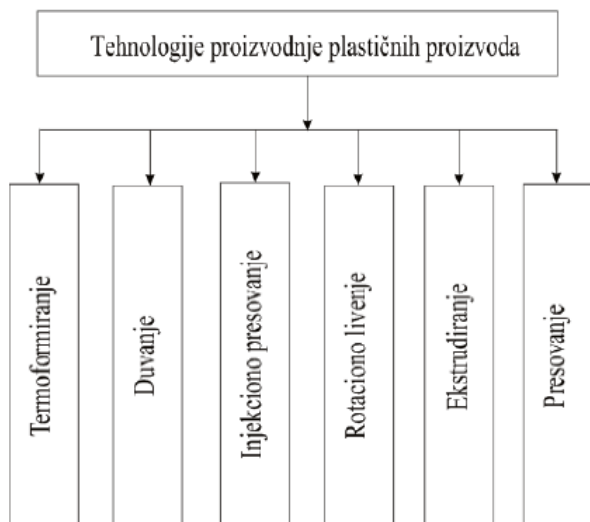


Slika 2.6 Upotreba polimera u medicini i farmaciji[10]

3. KLASIFIKACIJA POSTUPAKA PRERADE POLIMERA

Savremeni proizvodni sistemi u mašinskoj industriji odlikuju se proširenjem asortimana proizvoda, visokom frekvencijom promene programa proizvodnje, zahtevima za stalnim poboljšanjem kvaliteta proizvoda, alata, smanjenjem rokova i troškova njihove izrade, stalnom potrebom podizanja tehnološkog nivoa proizvoda itd. Jedna od najvažnijih karakteristika savremenog proizvodnog sistema jeste sposobnost da se u što kraćem vremenu projektuje i proizvede mnoštvo visokokvalitetnih proizvoda. Brzo lansiranje novog proizvoda na tržište, pre nego što to učini konkurencija, predstavlja ključni faktor u obezbeđivanju većeg dela tržišta i viših profitnih stopa. Ključni aspekt u razvoju proizvoda jesu faze konceptijskog i detaljnog projektovanja u kojima treba da dođe do stvaranja i vrednovanja ideja za novi proizvod. [1]

Podela tehnologija za proizvodnju proizvoda od plastike prikazana je na slici 3.1.

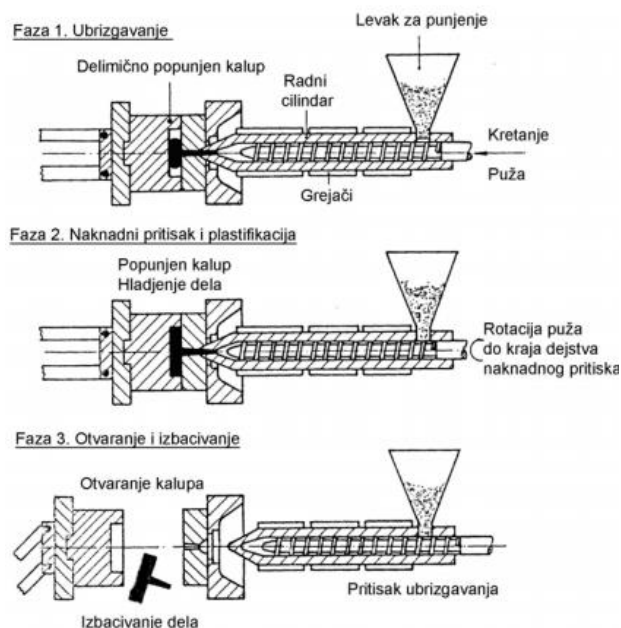


Slika 3.1 Tehnologije prerade polimera[1]

U nastavku teksta opisani su postupci koji se najčešće upotrebljavaju.

Injekciono presovanje je najvažniji ciklični postupak prerade polimera, a prema dostignutoj tehnologiji i najusavršeniji. Ovaj postupak spada u primarnu preradu polimera tj. u praoblikovanje polimera, jer se oblik otpreska dobija od polaznog materijala koji nema određenu formu (granule, komadići i sl.) Injekcionim presovanjem se oblikuju svi polimeri: duromeri, elastomeri, elastoplastomeri, a posebno je raširena prerada plastomernih materijala. Može se smatrati da je injekciono presovanje polimera slično livenju pod pritiskom metala. Injekcionim presovanjem se osim polimera prerađuju i keramičke smeše, kombinacije različitih materijala (npr. plastika, metal i keramika) i biomaterijali. Ovaj postupak se izvodi ubrizgavanjem rastopljenog polimera određene viskoznosti iz injekcione jedinice u temperiran kalup. Kratko vreme ciklusa, visoka ponovljivost kvaliteta dobijenih otpresaka, visoka dimenziona tačnosti i velika raznolikost otpresaka različite složenosti, čini tehnologiju injekcionog presovanja vrlo popularnom u preradi plastike. Glavne prednosti ovog postupka su u velikoj mogućnosti

iskorišćenja plastičnog materijala, manjem vremenu izrade proizvoda, visokoj dimenzionoj tačnosti otpreska, i manjem potrebnom prostoru za proizvodnju. [5]



Slika 3.2 Šema procesa injekcionog presovanja[5]

Ekstrudiranje je kontinualni postupak prerade plastičnih masa. Ovim postupkom plastična masa, kao polazna sirovina, u prahu ili što je najčešće u granulatu raznih oblika ubacuje se putem levka u cilindar mašine u kojoj je smešten jedan ili više puževa koji plastičnu masu transportuju, a pod uticajem dovedene toplote prevode je u tečno stanje. Dejstvom pogona za obrtanje puža kao i savlađivanjem otpora koji nastaju transportovanjem rastopljene plastične mase, kroz otvore između puža i cilindra masa se plastificira, homogenizuje i na kraju u alatu mašine formira u željeni oblik. Ova mašina u kojoj se odvija pomenuti proces prerade plastike zove se ekstruder i prikazana je na slici 3.3.



Slika 3.3 Jednupužni ekstruder[5]

Jednupužni ekstruder sastoji se od:

- 1) puža, 2) cilindra, 3) spojnica za spajanje sa alatom, 4) vođeno hlađene ulazne zone, 5) levka za doziranje sirovina, 6) pogona ekstrudera, 7) temperiranje puža.

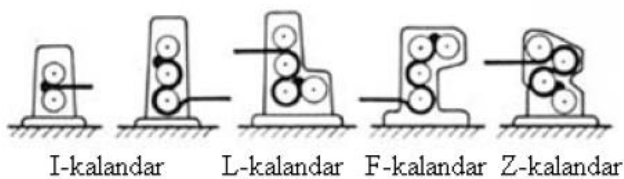
Ekstruderi sa jednim pužem imaju veliku primenu u preradi polimera, a prednost ovih mašina u odnosu na druge su:

- niža cena,
- jendostavna, robusna i pouzdana konstrukcija,
- dobre tehničke performanse

Ekstrudiranjem se u farmaceutskoj industriji uobičajeno izrađuju bočice manjih dimenzija kao što je bočica u kojoj se nalaze kapi za oči.

Kalandriranje je kontinualan postupak izrade beskonačnih traka praoblikovanjem visoko viskoznog rastopljenog polimera njegovim propuštanjem između parova valjaka kalandra s prilagodljivim zazorom. Obradak, kalandrat u obliku traka nastaje očvršćivanjem procesima geliranja i hlađenja, hlađenja ili umrežavanja. Kalandriraju se smeše od prirodnog i sintetičkog kaučuka i plastomera. Kalandriranje plastomera počelo je najpre sa smešama poli(vinil-hlorida), što je i danas najčešće prerađivani polimer tim postupkom (60 % svih kalandrata) [5]

Najvažniji element linije za kalandriranje jeste kalandar. Osnovna uloga kalandra je da delovanjem pritiska razvalja na potrebnu debljinu dobavljeni prethodno omekšani polimer. Kalandri se razlikuju prema broju valjaka i njihovom međusobnom položaju. Prema broju valjaka mogu biti: dvovaljčani do peterovaljčanih, a prema položaju valjaka razlikuje se I, F, L, Z i S izvedba, što se vidi na slici 3.4



Slika 3.4 Kalandar

4. ZAKLJUČAK

Kada je u pitanju primena u medicini može se zaključiti da polimerni biomaterijali pokazuju značajne prednosti u odnosu na ostale materijale. Prednosti polimernih materijala su u tome što su vrlo biokompatibilni, mogu se lako izraditi u proizvode raznih oblika, mogu se reciklirati i imaju povoljne mehaničke i fizičke osobine. Od velike je važnosti da izrada i obrada pomenutih materijala bude precizna i čista, kako bi se mogućnost za neke poremećaje u organizmu, npr. trovanje i slično svele na najmanju moguću meru. Takođe, bitno je da tokom obrade ne dolazi do hemijskih ili mehaničkih promena u samom materijalu. Polimerni materijali, kao i svi drugi materijali imaju svoje prednosti i nedostatke pa samim tim imaju i ograničenost u primeni. Medicina je opširno područje, a zastupljenost polimernih materijala u medicini je vrlo velika. Kako se medicina razvija tako se i zahtevi prema polimernim materijalima sve više povećavaju. U radu je napravljen pregled polimernih materijala koji su najviše i najčešće u upotrebi, a to su oni materijali koji svojim svojstvima najbolje odgovaraju sve strožijim postavljenim zahtevima sa aspekta sigurnosti, biokompatibilnosti i hemijske postojanosti. Proizvodi od plastike danas znatno pridonose kvalitetu života te očuvanju prirodnih resursa u klime, stoga je plastika važna u industriji. Svakodnevno veliki broj stručnjaka radi na razvoju novih materijala koji

imaju sve bolja mehanička i termička svojstva, duži vek trajanja te manji uticaj na okolinu. Tako su sigurnosni i zdravstveni zahtevi te zahtevi zaštite okoline doveli do razvoja novih nanomaterijala, biorazgradljivih te biokompatibilnih polimernih materijala. Razvijaju se novi postupci prerade polimera, te se poboljšavaju postojeći preradbeni postupci u svrhu poboljšavanja kvaliteta proizvoda, smanjenja potrošnje prirodnih izvora sirovina te očuvanja okoline. Ekonomski zahtevi, zakoni o zaštiti okoline, niski troškovi proizvodnje, skraćanje vremena od dizajna proizvoda pa do njegovog izlaska na tržište, zahtevi su koji teraju proizvođače da proizvode kvalitetnije, jeftinije i brže. Zbog tih sve većih zahteva proizvođači nastoje koristiti materijale, proizvodne postupke i opremu koja je u skladu sa zadatim zahtevima te koja će skratiti proizvodni ciklus, proizvoditi kvalitetnije proizvode, imati manji utrošak energije, manje zagađivati okolinu, a sve će to rezultirati smanjenjem cene samog proizvoda.

5. LITERATURA

- [1] Matin, I.: Modularni sistemi za projektovanje alata za injekciono presovanje plastike, Univerzitet u Novom Sadu, Fakultet tehničkih nauka, Novi Sad, 2011.
- [2] Pavlinec, M.: Razvoj plastične ambalaže za pakovanje farmaceutskog proizvoda, Sveučilište u Zagrebu, Fakultet strojarstva i brodogradnje, Zagreb, 2017.
- [3] Stojiljković, D., Pilić, B.: Struktura i svojstva polimernih materijala, Univerzitet u Novom Sadu, Tehnološki fakultet, Novi Sad, 2007.
- [4] Savić, V.: Osnovni mehanizmi hlađenja mašina za prizganje plastike pod pritiskom, Požega, 2009.
- [5] Vilotić, D. Tehnologija injekcionog presovanja polimera, Univerzitet u Novom Sadu, Fakultet tehničkih nauka, Novi Sad, 2007.
- [6] <http://ru.goldenwellgermany.com/disposable-syringe/irrigation-syringe/plastic-100ml-irrigation-syringe>
- [7] <http://dental4u.ba/aktuelnosti-iz-stomatologije/clanci/peek-polyether-ether-ketone/>
- [8] <https://www.termoplast.rs/tehnicka-plastika/termoplast-peek>
- [9] <https://www.dental-medical.rs/blog/peek-vs-pekk>
- [10] <https://www.grandviewresearch.com/industry-analysis/medical-plastics-market>
- [11] <http://srla.fukang-jars.com/plastic-jar/shenzhen-port-pet-pill-bottle-storage-in.html>

Kratka biografija:



Vanja Vidović rođena je u Vlasenici 1994. godine. Diplomirala na Fakultetu tehničkih nauka smer Proizvodno mašinstvo 2018. god. Master studije upisala iste godine na usmerenju Savremene tehnologije oblikovanja plastike.

Mladimir Milutinović rođen je u Arilju 1967. godine Doktorirao je na Fakultetu tehničkih nauka 2013. god., iste godine izabran je u zvanje docenta. Oblast interesovanja su tehnologije plastičnog deformisanja, aditivne i virtuelne tehnologije

IZBOR MERODAVNIH REŽIMA OBRADE PRI IZRADI KUĆIŠTA VENTILA ITRON

SELECTION OF RELEVANT PROCESSING REGIMES IN THE MANUFACTURE OF VALVE HOUSING ITRON

Ana Vujinov, Borislav Savković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj – U ovom radu prikazan je proces prikupljanja podataka parametara režima obrade za određene reference koje se proizvode u preduzeću Mecafor, zbog trenutnog i sistematičnog uzimanja režima obrade. Izvršen je monitoring procesa obrade pri bušenju i glodanju u cilju smanjenja vremena obrade odnosno povećanja cilja, tj. broja obrađenih komada. Takođe je prikazno idejno i krajnje rešenja pribora, za kućište ventila Itron.

Ključne reči: *Režimi obrade, optimizacija, projektovanje steznog pribora*

Abstract – This paper presents the process of collecting the parameters of the machining regime for specified references produced by Mecafor, due to the current situation and systematic machining regime acquisition. The drilling and grinding process were monitored to reduce processing time and increase the number of processed pieces. For the ITRON valve housing, the conceptual and final solution of the clamping accessories is also shown.

Ključne reči: Processing regimes, optimization, Designing fixture

1. UVOD

U doba strahovito brzog razvoja tehnologije, mašinska industrija, kao i sve ostale oblasti teži ka svom idealu. Pod idealom se u proizvodnji podrazumeva najbolje moguće iskorišćenje materijalnih i nematerijalnih resursa. Vreme, novac, uloženi rad, samo su neki od resursa koji se uz pomoć tehnologije današnjice mogu mnogo bolje iskoristiti u cilju poboljšanja produktivnosti, kao i profita preduzeća. Savremeni proizvodni sistemi u mašinskoj industriji odlikuju se:

- proširenjem asortimana proizvoda,
- visokom frekvencijom promene programa proizvodnje,
- zahtevima za stalnim poboljšanjem kvaliteta proizvoda,
- smanjenjem rokova,
- smanjenje troškova izrade,
- stalnom potrebom podizanja tehnološkog nivoa proizvoda itd.

Jedna od najvažnijih karakteristika savremenog proizvodnog sistema jeste sposobnost da se u što kraćem

vremenu projektuje i proizvede mnoštvo visokokvalitetnih proizvoda [1-3].

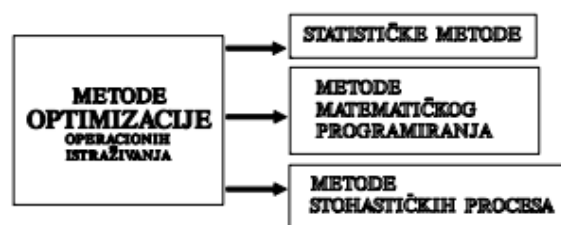
Brzo lansiranje novog proizvoda na tržište, pre nego što to učini konkurencija, predstavlja ključni faktor u obezbeđivanju većeg dela tržišta i viših profitnih stopa.

2. OPTIMIZACIJA PROCESA OBRADE

Optimizacija je postupak nalaženja najpovoljnijeg rešenja konstrukcije pri zadatim uslovima. U teoriji optimalnog projektovanja, optimizacijom se određuju konstruktivni parametri (geometrija) koji definišu ekstremna svojstva (minimum-maksimum) posmatranih mašina.

Optimizacija je u matematičkom smislu, proces nalaženja uslova koji daju ekstremne vrednosti funkcija cilja.

Optimizacija je primenjena naučna disciplina koja metodama matematičkog programiranja, varijacionog računa, teorijom optimalnog upravljanja i metodama teorijske mehanike, definiše tražena tehnička svojstva konstrukcija. Jedna opšta klasifikacija metoda optimizacije operacionih istraživanja:



Slika 1. Metode optimizacije u širem smislu

3. REŽIMI PROCESA OBRADE

Pod režimom procesa obrade se podrazumevaju svi oni parametri koje treba podesiti na mašini da bi data obrada mogla nesmetano da se obavi [3-5]. To su oni parametri kojima se definišu glavno, pomoćno, pa i neka od sporednih kretanja pri obradi [5-7]. Režime obrade sačinjavaju:

- Dubina rezanja: a ,
- Pomak: s ,
- Brzina rezanja: v .

Dubina rezanja, a

- je ona vrednost za koju se pomakne alat ili obradak da bi alat došao u poziciju skidanja sloja materijala.

Pomak, s

- je ona veličina za koju se alat ili obradak pomakne za jedan okretaj glavnog vretena

- daje širinu strugotine

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Borislav Savković.

- zavisi od vrste obrade i kvaliteta obrađene površine (za grubu obradu je najveći, za finu manji)

Brzina rezanja, v

- je obodna brzina kojom oštrica odvaja česticu
- ima je alat pri glodanju i busenju, odnosno obradak pri struganju
- merna jedinica je m/min, m/s

Praćenje ovog procesa sastoji se iz više faza, u prvoj fazi vrši se kontinualni nadzor praćenja grešaka pri obradi mašinskih odlivaka, promena parametara zbog optimizacije procesa a sve radi smanjenja vremena obrade [3,4]. Sa povećanjem broja obrađenih komada raste i produktivnost s tim da se mora izvršiti i monitoring intenzivnog habanja alata i pločica [2,5,6]. Zatim se ove informacije unose u posebne tabele koje su formirane radi izračunavanja:

- vremena obrade (pri različitim režimima obrade),
- vremena pripreme,
- izračunavanje broja obrađenih komada, radi lakšeg pronalazenja grešaka pri povećanim ili smanjenim unetim vrednostima iz programa obrade.

Radi lakšeg prikaza u nastavku je izdvojen samo segment tehnologije obrade za jednu referencu, odnosno za kućište ventila tipa ITRON. Na slici 2. dat je prvobitni prikaz operacija sa korišćenim režimima obrade za odgovarajuće alate.

Referenca	Mašina	Vreme obrade	Vreme pripreme	Datum merenja	Broj komada	Oznaka alata	Vrsta alata	Prečnik alata	S obrt/min	F mm/min	Broj zuba	Vc m/min	fz mm/zub
722	104	00:10:05	00:00:55	16-01-19	44.2	T2007A	Glodalo	63	2500	1000	8	495	0.05
722					91%	T3145	Burgja	19	5500	700	1	109	0.20
722						T1616	Glodalo	16	3500	700	6	176	0.03
722						T4545	Glodalo	26	1300	500	4	106	0.10
722						T2006	Glodalo	20	1300	100	4	160	0.03
722						T68	Busenje	6.8	3500	700	1	75	0.20
722						M8	Ureznik	8	1000	1000	1	25	1.00

Slika 2. Prikaz podataka iz baze obrade

$$N = \frac{480}{\left(\left(60 * \text{Ukupno vreme obrade} + \frac{\text{Ukupno vreme obrade}}{60} \right) \right)} + \left(\left(60 * \text{Vreme operatora} + \frac{\text{Vreme operatora}}{60} \right) \right) * \text{Procentualno iskorišćenje mašine} = [\text{kom/smeni}] \quad (1)$$

Minimalno iskrošćenje mašine u proizvodnom pogonu je 85% i ispod te vrednosti se smatra da je proizvodnja ne produktivna. Na osnovu prikazane analize prikazane na slici 2, vidi se da je moguća korekcija režima obrade na

pozicijama kod burgje T3145 i glodala T1006. Ova korekcija se vrši na osnovu preporučenih režima obrade za od proizvođača za burgiju prečnika $\phi 19$, prikazanog na slici 3.

Vrsta materijala	Hladjenje	Proizvodja	Ident. broj burgje	Upotreba (referenca)	Teoretska Vc (m/m)	Teoretski f (mm/d)
TM	H	Seco Tools	1526	GGP 82034039	75	0.35
Broj al	Radna burgija	Vrsta burgje	Oznaka burgije	Oznaka umetka (pločica)	D	L
T3145	T3145	N	SD105-19.00-19.99-95-20R7	SD100-19.84-P	$\phi 19$	SD

Slika 3. Prikaz podataka burgije T3145 od strane proizvođača

Prizvođač je dao preporučenu teoretsku minimalnu brzinu obrade ispod koje se ne preporučuje obrada. Na osnovu praćenja habanja alata u radu, ustanovljeno je da moguće kontinualno povećanje brzine sve dok se ne dođe do kritične zone u kojoj nastaje oštećenje alata. Konkretno za ovaj primer brzina rezanja u odnosu na zatečeno stanje je povećana za 40%. Nakon korekcije režima obrade formirana je nova tabela, slika 4. Gde se vide novi režimi obrade kao i novo ukupno vreme obrade kućišta ventila

ITRON. Na osnovu jednačine 1, povećana je proizvodnost u jednoj smeni sa 44 kom za osam sati rada na 47. U novoj analizi predviđen je veći procenat iskorišćenja, odnosno sa 85% je povećan na 91%. U nastavku je prikazan deo baze reznih alata za bušenje pristnih u proizvodnom pogonu firme Mecafor, gde se mogu videti osnovne karakteristike jednog alata a to su:

722	104	00:09:18	00:00:45	16-01-19	47.7	T2007A	Glodalo	63	2500	1000	8	495	0.05
722					91%	T3145	Burgija	19	3500	850	1	150	0.24
722						T1616	Glodalo	16	3500	700	6	176	0.03
722						T4545	Glodalo	26	1300	500	4	106	0.10
722						T1006	Glodalo	20	1500	200	4	200	0.03
722						T68	Busenje	6.8	3500	700	1	75	0.20
722						M8	Ureznik	8	1000	1000	1	25	1.00

Slika 4. Prikaz korigovane baze podataka

- Redni broj alata po kojem se izdaje,
- Vrsta burgija koja može biti: N-standardna burgija, S- specijalna burgija,
- Oznaka burgije (izdaje proizvođač),
- Oznaka pločice (izdaje proizvođač),
- Prečnik alata,
- Dužina alata,
- Vrsta materijala (za koju je predviđena obrada),
- Hlađenje (sa ili bez hlađenja),
- Proizvođač alata,
- Identifikacioni broj burgije (zadaje zaprimalac u preduzeću),
- Upotrebna referenca,
- Teoretski Vc (m/min), (zadaje proizvođač alata),
- Teoretski f (mm/o), (zadaje proizvođač alata)

1	Broj al	Radna burgija	Vrsta burg	Oznaka burgije	Oznaka umetka (pločica)	D	L	
9	T3004	T3004	N	A3999KPL-13.7		13.7	6D	
10	T3004	T3004	N	A3985TFL-13.7		13.7	6D	
11	T3004	T3004	N	05511-13,700 N DIN 6537L R-RT1		13.7	6D	
12	T3005	T3005	U	80109401.0060		M8		
13	T3006	T3006	N	A3285TFL-7.7		7.7	3D	
1	Broj al	Vrsta materijala	Hlađenje	Proizvođa	Ident.broj burg	Upotreba (referenca)	Teoretska Vc (m/m	Teoretski f (mm/c
9	T3004	TM	H	Walter	133	Arvin Meritor	90	0.25
10	T3004	TM	H	Walter	133	Arvin Meritor		
11	T3004	TM	H	Gühring	133	Arvin Meritor	100	0.3
12	T3005	HSS	N	Emuge	7	Palier 261	19	2

Slika 5. Baza alata za bušenje

Ovakve baze podataka postoje i za operaciju obrade glodanjem.

Na osnovu analize baze podataka uviđa se da su u proizvodnom pogonu u upotrebi specijalni i standardni alati, kako burgije tako i glodala.

Specijalna burgija od tvrdog metala sa hlađenjem za visoko efikasno bušenje je primerena za veće vrednosti pomaka. Karakteristike burgija su dvostruka vodeća oštrica, koja garantuje tačnost promera, talasastu rezu oštricu za manju silu rezanja, efikasno odvođenje strugotine i nova AlCrTiN-prevlaka Sumi-Power, otporna na stvaranje naslaga.

Primer upotrebe ovog tipa burgije u radu za prečnik 8 mm, vc= 80 m/min, f= 0,15 mm/obr, za prolazno bušenje sa hlađenjem, donosi rezultat od 2000 izrađenih otvora s tim da nije došlo do habanja alata, odnosno primena ove burgije se može nastaviti.

S druge strane u proizvodnom pogonu su prisutne i standardne burgije, koje imaju postojanost od 100 do 130 metara, odnosno oko 125 radnih časova. Primenom novih specijalnih burgija produktivnost je povećana i do 25%.

Tendencija primene specijalnih burgija u proizvodnom pogonu je u porastu. Da je to dobra odluka za prelazak na nove specijalne alate potkrepljeno je činjenicom da specijalne burgije od HSS-a visokog kvaliteta predviđene za legirane i nelegirane čelike, nerđajuće čelike, liveno gvožđe, aluminijum, aluminijumske legure, bakar, mesing i bronzu, pri testiranju na obradi mašinskog odlivka od sivog liva kao i nodularnog liva imaju radni vek u proseku deset radnih dana dok je za standardnu burgiju ta vrednost iznositi pet radnih dana.

U nastavku su prikazane “ Pivot Table ” koje predstavljaju bazu koju sami formiramo, služe za jasan uvid i brzu pretragu alata i režima za odgovarajuće reference koje su ređene u proizvodnom pogonu.

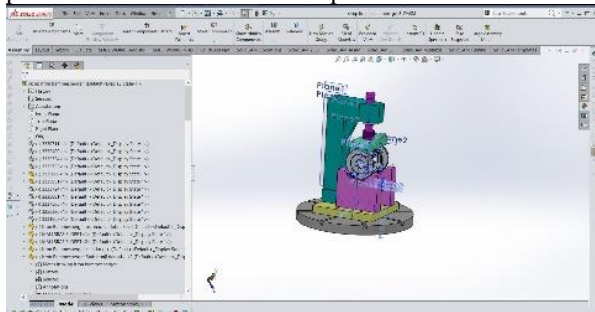
Tehnolog prema preporukama daje akcenat na režime koji su povoljni za obradu i odnosno koi ne donose dobre rezultate. U Pivot tabelama žutom bojom su predstavljene reference gde je potrebna korekcija.

processing technology	Tool diameter Ø mm	Tool	S Speed / min	F Pos. Mm / min		
out of Vc - m/min						
ehnologija obrade	Prečnik alata Ø mm	Alat	S Broj obrtaja / min	Vc - m/min	fz mm/z	F Posmak mm/min
Busenje otvora/drilling						Referenca Total
	2.50	T38	7639	53.96615	0.119910893	916
			7639	53.96615	0.119910893	916
	3.2	T69	7420	74.55616	0.150643396	1120
	5.00					3
	6.80	T3034	4200	90	0.3	1090
		4890	104.41128	0.2	1200	489
		4900	104.6248	0.2	1090	942
				0.3	1300	489
		7493	159.99036	0.199919925	1498	942
	T3070	4200	89.6784	0.25962381	1090	968
	T54	7490	159.92648	0.200267023	1500	486
	T59	7000	149.464	0.164285714	1150	486
	T68	3000	64.056	0.2	600	469
	T6885	5000	106.76	0.24	1200	674/675
	T842 6.8	7022	149.93744	0.04	1752	352
		7023	149.95096	0.250075587	1756	353
	T902	7960	169.96192	0.270309569	2150	516
	T3142	4000	85.408	0.2325	850	675
	T18	5992	159.92648	0.083444693	500	486
		5992	159.92648	0	1696	516
	T58	7490	159.9081	0.2	1498	486
	T8511	5600	149.464	0.303571429	1700	OMP
	T3028	4200	119	0.3	1245	943
	T84	3600	101.736	0.222222222	800	516
	T840 9.1	5247	149.92778	0.06667902	2099	352
	T842 9.1	5552	158.642848	0.300072046	1666	352
		6250	176.5875	0.272	1700	353
				0.33584	2099	353
	10.20	T842 10.2	121.7694	0.229473684	872	353
				0.018157895	870	352
		T3051	3000	96.034	0.266666667	900

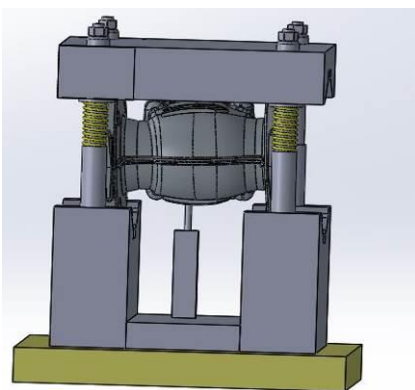
Slika 6. Prikaz iz "Pivot table" za obradu bušenjem

4. PROJEKTOVANJE PRIBORA

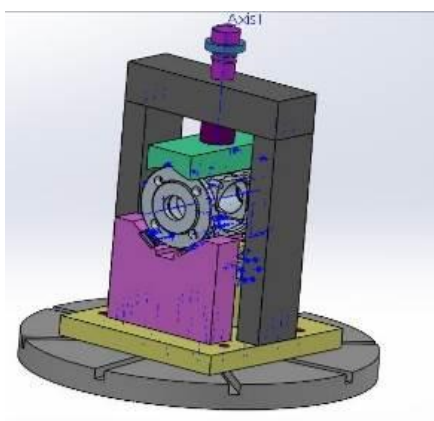
Procesi transformacije polufabrikata u gotove proizvode i poluproizvode odvijaju se kompleksnim dejstvom sredstava za rad na materijal odnosno radni predmet. Do traženog oblika ili gotovog proizvoda dolazi se po definisanom redosledu pojedinih operacija i zahvata, po predviđenom tehnološkom postupku izrade. U svakom obradnom procesu, pored radnog predmeta, alatne mašine, alata i merne instrumentacije, neminovno je da, kao materijalni činilac lanca, bude prisutan i pribor. U nastavku su dati primeri varijanti izvedbi steznih pribora za obradu kućišta kompresora Itron.



Slika 7. Prikaz prvobitnog - odbačenog idejnog rešenja



Slika 8. Prikaz drugog - odbačenog idejnog rešenja



Slika 9. Prikaz prihvaćenog idejnog rešenja za kućište ventila

Krajnje rešenje nam daje sledeće prednosti u odnosu na ostale izvedbe:

- Veća bezbednost pri radu,
- Lako postavlja i skidanje komada,
- Veća fleksibilnost.

5. ZAKLJUČAK

Analiza stanja sistema nam daje jasan uvid u kojoj se fazi nalaze alati i režimi koji su prisutni u proizvodnom pogonu. Ovim postupkom moguće je unaprediti proizvodni sistem, odnosno moguće je smanjiti broj alata, skratiti vreme obrade, a samim tim i povećati produktivnosti.

Formiranjem baze, uz redovno praćenje procesa, moguće je na nedeljnom nivou, korigovanje programa obrade sve dok ne postigne odgovarajuće poboljšanje. Baza podataka omogućava dodavanje, i nadogradnju, zatim poređenje sa predhodnim ciljevima. Jasan uvid daje dobre rezultate, pogotovo u novim preduzećima, gde je cilj samo obrađivati komade bez obzira na mogućnost iskorišćenja svih mogućih resursa, i procesa optimizacije.

Pre početka svakog rada, tako i pri analizi i projektovanju potrebno je sagledati prednosti i nedostatke koje se mogu desiti novim idejnim rešenjem. Analiza obrade donosi dobrobit preduzeću jer daje jasan uvid na trenutnu situaciju, a njenim razvojem mogu se postići znatne prednosti, kao što je prikazano u primerima. Izborom adekvatnog alata, na određenoj mašini sa steznim priborom omogućavaju produktivnu obradu. Imajući više varijanti steznog pribora lako se sagledavaju prednosti i nedostaci. Samim tim eliminisanjem ne adekvatnog rešenja sve smo bliže pravom rešenju.

6. LITERATURA

- [1] Kovač P., Milikić D.: Rezanje metala, „univerzitetski udžbenik“ Univerzitet u Novom Sadu, Fakultet Tehničkih nauka, 1998.
- [2] Kovač P.: Teorija obradnih procesa, Univerzitet u Novom Sadu, Fakultet Tehničkih nauka, 2014.
- [3] Kovač P., Milikić D., Gostimirović M., Savković B., Sekulić M.: Tehnologija obrade rezanjem, Zbirka rešenih zadataka, Univerzitet u Novom Sadu, 2013.
- [4] Nedić, B., Lazić M.: Obrada metala rezanjem, Mašinski fakultet- Kragujevac
- [5] Stanić, J., Teorija procesa obrade I, Mašinski fakultet, Beograd, 1995.
- [6] Nikolić, D., Teorija procesa obrade II, Mašinski fakultet, Beograd, 1998.
- [7] Ekinović, S.: Obrada rezanjem, Dom štampe, Zenica, 2001

Kratka biografija:



Ana Vujin rođena je u Zrenjaninu 1995.god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Proizvodnog mašinstva odbranila je 2018.god.



Borislav Savković rođen je u Novom Sadu 1982. god. Doktorirao je na Fakultetu tehničkih nauka 2015. god. gde je i zasnovao radni odnos u zvanju docenta. Autor je jednog univerzitetskog udžbenika i preko 100 naučnih radova. Oblast interesovanja su procesi obrade skidanjem materijala, simulacije kao i ekološko tehnološki sistemi.

TRIBODIJAGNOSTIKA STATIČKOG KOEFICIJENTA TRENJA KONTAKTNIH PAROVA OD DRVETA**TRIBODIAGNOSTICS OF THE STATIC FRICTION COEFFICIENT FOR WOOD FRICTION PAIRS**

Nikola Đokić, Đorđe Vukelić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – PROIZVODNO MAŠINSTVO

Kratak sadržaj – U radu su prezentovani rezultati merenja statičkog koeficijenta trenja kontaktnih parova od drveta. Istraživanja su sprovedena na tribometru koji funkcioniše po principu kose ravni. Dobijene srednje vrednosti statičkog koeficijenta trenja za kontaktne parove od drveta se kreću u rasponu od 0,42 do 0,63. Veće vrednosti statičkog koeficijenta trenja su dobijene za kontaktne parove od drveta koje karakteriše veća tvrdoća i veći modul elastičnosti, i obrnuto.

Ključne reči: Tribometar, statički koeficijent trenja, drvo

Abstract – This paper presents the results of measuring the static friction coefficient for wood friction pairs. Research has been carried out on a tribometer that operates on the principle of an inclined plane. The obtained mean values of the static friction coefficient for wood contact pairs range from 0.42 to 0.63. The higher values of the static friction coefficient are obtained for wood friction pairs characterized by higher hardness and a higher modulus of elasticity, and vice versa.

Keywords: Tribometer, static friction coefficient, wood

1. UVOD

Drvo je prirodan materijal koji karakteriše niz pozitivnih svojstava, kao što su: lakoća obrade, prihvatljiva cena, niska toplotna provodljivost, itd. Da bi se izvršila predikcija ponašanje drveta u različitim uslovima eksploatacije, potrebno je poznavati njegove tehnološke, mehaničke i fizičke karakteristike. Poznavanje triboloških svojstava jedan je od preduslova za racionalnu eksploataciju kontaktnih parova napravljenih od drveta [1,2].

Trenje je složen fenomen koji nastaje pri kontaktu površina. Procesi trenja kontaktnih parova odvijaju se u složenim uslovima a istraživanja su pokazala da postoji funkcionalnu zavisnost trenja od velikog broja parametara, kao što su: materijali i geometrija kontakta, kontakti pritisci, temperatura, opterećenje, vlažnost, stanje kontaktne površine, itd. [3,4].

U većini slučajeva, u različitim sistemima trenje je nepoželjna i štetna pojava. Na savlađivanje sile trenja troši se određena količina energije a kao posledica trenja na površinama u kontaktu dolazi do trošenja i zagrevanja materijala.

Tribološke karakteristike elemenata sistema mogu se kvantifikovati intenzitetom sile trenja odnosno koeficijentom trenja [5]. Sila trenja je proporcionalna normalnoj sili koja deluje na površine koje su u kontaktu, što se može prikazati sledećim izrazom [6]:

$$F_t = \mu \cdot F_n \quad (1)$$

gde je: F_t sila trenja, μ koeficijent trenja a F_n normalna sila.

Veličina sile trenja ne zavisi od veličine kontaktne površine i ravnomerno je raspoređena po njoj. Sila trenja deluje tangencijalno u odnosu na kontaktnu površinu i ima smer dejstva suprotan smeru tendencije kretanja [6].

Objavljeno je nekoliko istraživanja o trenju kontaktnih parova od drveta. Blau [6] je predstavio koeficijente trenja za različite vrste drveta. Vrednosti statičkog koeficijenta trenja su se kretale u rasponu od 0,25 do 0,7, a kinematskog koeficijenta trenja u rasponu od 0,19 do 0,5. Glass i Zelinka [7] su merili koeficijente kinematskog koeficijenta trenja za različite vrste drveta. Koeficijenti trenja pri srednjem sadržaju vlage su bili u rasponu od 0,3 do 0,7. Pitenis i saradnici [8] su merili koeficijente trenja za drvo za različite mikroklimatske uslove. Vrednosti statičkog koeficijenta trenja su iznosile od 0,35 do 0,72. Aira i saradnici [9] su objavili rezultate statičkih i kinematskih koeficijenata trenja za meko drvo (crni bor). Xu i saradnici [10] merili su koeficijente trenja drveta sa različitim kvalitetom kontaktnih površina. Rezultati ukazuju da se koeficijenti trenja drveta linearno povećavaju sa srednjom aritmetičkim hrapavošću profila površine (R_a). Sathre i Gorman [11] su određivali statičke i kinematske koeficijente trenja kontaktnih parova od drveta javora bez prisustva maziva između kontaktnih površina. Koeficijent statičkog trenja imao je vrednost od oko 0,45, dok je koeficijent kinematskog trenja imao vrednost od oko 0,35.

Nadalje, Coulombo je izmerio koeficijente statičkog trenja klizanja u rasponu od 0,43 do 0,67 za različite uzorke neobrađenog drva. Ukoliko se vrši podmazivanje kontaktnih površina lojem vrednosti koeficijenta trenja se smanjuju na nivo 0,1-0,2. Za klizanje drveta po čeliku, Bovden i Tabor su izmerili statički koeficijent trenja od 0,6, a Rabinovicz u rasponu od 0,45 do 0,5 [5,12].

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Đorđe Vukelić, vanredni profesor.

Za razliku od predhodnih, cilj ovog istraživanja jeste da se izmere statički koeficijenti trenja klizanja uzoraka od različitih vrsti drveta.

2. METODOLOGIJA

Tribometar (slika 1) upotrebljen u eksperimentalnim istraživanjima koristi se za ispitivanje statičkog koeficijenta trenja različitih vrsta materijala.



Slika 1. Tribometar

Uređaj funkcioniše po principu kose ravni. Meri se ugao nagiba kose ravni pri kome dolazi do kretanja jednog bloka u odnosu na drugi. Uređaj poseduje dve skale. Na jednoj skali je baždaren ugao nagiba kose ravni (α) a na drugoj su baždarene vrednosti statičkog koeficijenta trenja (μ).

Statički koeficijent trenja (μ) je bezdimenziona karakteristika trenja. Po definiciji jednak je količniku sile trenja (F_t) i normalne sile (F_n). Merenje statičkog koeficijenta trenja, po principu kose ravni, zasniva na gravitacionoj sili ($m \cdot g$) koja deluje na telo mase (m) koje je kosoj ravni koja je nagnutu u odnosu na horizontalu za određeni ugao (α). U graničnom slučaju trenja klizanja važe jednačine:

$$\mu = F_t / F_n \quad (2)$$

$$F_t = m \cdot g \cdot \sin \alpha \quad (3)$$

$$F_n = m \cdot g \cdot \cos \alpha \quad (4)$$

$$\mu = m \cdot g \cdot \sin \alpha / m \cdot g \cdot \cos \alpha \quad (5)$$

$$\mu = \sin \alpha / \cos \alpha \quad (6)$$

$$\mu = \tan \alpha \quad (7)$$

To znači da je statički koeficijent trenja jednak tangensu ugla nagiba kose ravni pri kome dolazi do relativnog kretanja elemenata u kontaktu.

Pre merenja vrši se priprema blokova - elemenata kontaktnog para. Blokove je potrebno očistiti (alkoholom i suvom krpom) kako bi se sa njih otklonile nečistoće. Takođe je neophodno izvršiti nivelaciju uređaja dovođenjem diska u nulti položaj.

U telo tribometra se postavi prvi blok. Zakretanjem diska tribometra blok se dovodi u nulti položaj. Ugao nagiba kose ravni je 0. Potom se na prvi blok postavi drugi blok i na taj način se formira kontaktni par. Površina kontaktnih parova je 18 x 50 mm.

Nakon toga se vrši zakretanje diska tribometra sve dok se ne uoči početak kretanja jednog bloka u odnosu na drugi. U tom momentu se vrši očitavanje ugla nagiba pri kome je došlo do relativnog kretanja jednog bloka u odnosu na drugi. Istovremeno se na gornjoj skali očitava vrednost statičkog koeficijenta trenja (slika 2).



Slika 2. Postupak merenja.

3. REZULTATI

Kontaktni elementi sa kojima je izvršeno merenje statičkog koeficijenta trenja izrađeni su od sledećih vrsti drveta: jela, bor, hrast i jasen.

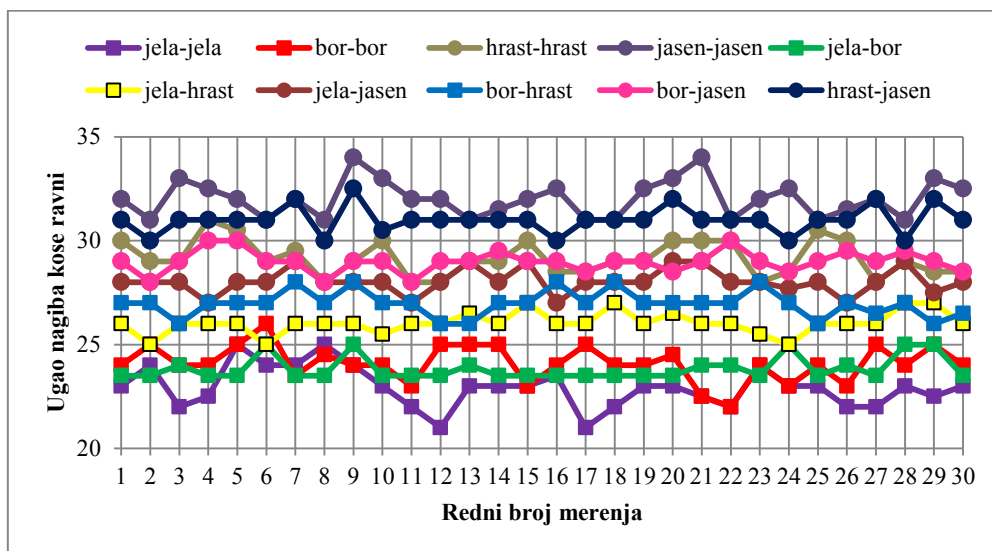
Pre početka istraživanja uzorci drvenih kontaktnih elemenata su sušeni u peći do sadržaja vlage <5% i brušeni na približno istu vrednost srednje aritmetičke vrednosti hrapavosti (R_a). Vrednosti fizičkih i mehaničkih karakteristika blokova upotrebljenih u eksperimentalnim istraživanjima su prikazane u tabeli 1.

Merenje ugla nagiba kose ravni izvršeno je za sledeće kombinacije materijala kontaktnih elemenata: jela - jela, bor - bor, hrast - hrast, jasen - jasen, jela - bor, jela - hrast, jela - jasen, bor - hrast, bor - jasen, i hrast - jasen.

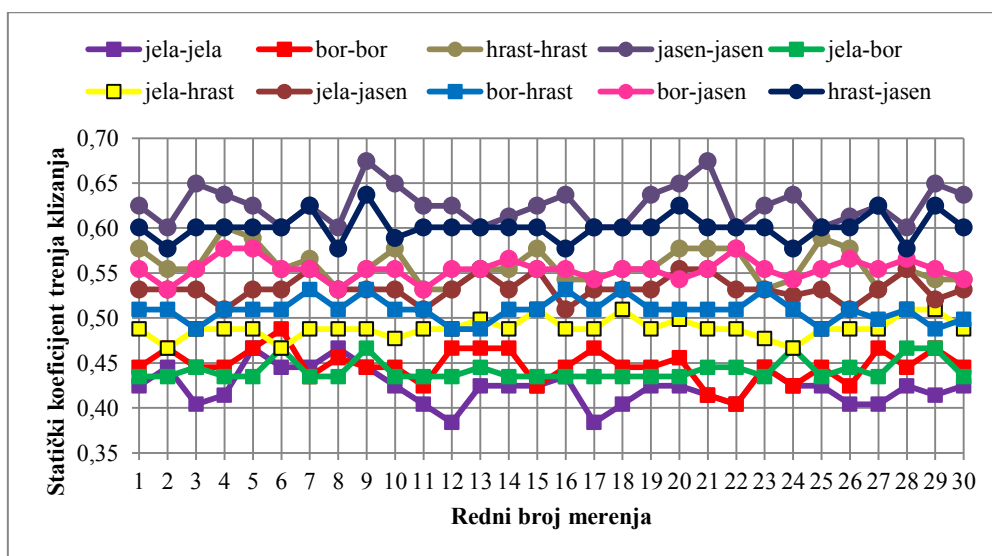
Za svaki kontakti par izvršeno je 30 merenja ugla nagiba kose ravni i statičkog koeficijenta trenja. Dobijeni rezultati su prikazani na slikama 3 i 4. Srednje aritmetičke vrednosti statičkog koeficijenta trenja, maksimalne vrednosti, minimalne vrednosti i standardna odstupanja su prikazani su u tabeli 2.

Tabela 1. Karakteristike materijala kontaktnih elemenata.

Karakteristika	Jela	Bor	Hrast	Jasen
Tvrdoća HB	39	35	75	76
Modul elastičnosti E (GPa)	11	12	12	14
Zatezna čvrstoća (N/mm ²)	86	112	102	182
Hrapavost Ra (μm)	1,63	1,64	1,62	1,61



Slika 3. Izmerene vrednosti ugla nagiba kose ravni.



Slika 4. Izmerene vrednosti statičkog koeficijenta trenja klizanja.

Tabela 2. Deskriptivna statistika dobijenih rezultata statičkog koeficijenta trenja.

Parametri	Kontakti par									
	jela-jela	bor-bor	hrast-hrast	jasen-jasen	jela-bor	jela-hrast	jela-jasen	bor-hrast	bor-jasen	hrast-jasen
Srednja aritmetička vrednost	0,42	0,45	0,56	0,63	0,44	0,49	0,53	0,51	0,55	0,60
Minimalna vrednost	0,38	0,40	0,53	0,60	0,43	0,47	0,51	0,49	0,53	0,58
Maksimalna vrednost	0,47	0,49	0,60	0,67	0,47	0,51	0,55	0,53	0,58	0,64
Standardna devijacija	0,02	0,02	0,02	0,02	0,01	0,01	0,01	0,01	0,01	0,01

Za različite kontaktne elemente od drveta izračunate su različite vrednosti statičkih koeficijenata trenja, i to za kontaktni par:

- jela-jela vrednosti kinematskog koeficijenta trenja se kreću u rasponu od 0,38-0,47;
- bor-bor vrednosti kinematskog koeficijenta trenja se kreću u rasponu od 0,40-0,49;
- hrast-hrast vrednosti kinematskog koeficijenta trenja se kreću u rasponu od 0,53-0,60;
- jasen-jasen vrednosti kinematskog koeficijenta trenja se kreću u rasponu od 0,60-0,67;
- jela-bor vrednosti kinematskog koeficijenta trenja se kreću u rasponu od 0,43-0,47;
- jela-hrast vrednosti kinematskog koeficijenta trenja se kreću u rasponu od 0,47-0,51;
- jela-jasen vrednosti kinematskog koeficijenta trenja se kreću u rasponu od 0,51-0,55;
- bor-hrast vrednosti kinematskog koeficijenta trenja se kreću u rasponu od 0,49-0,53;
- bor-jasen vrednosti kinematskog koeficijenta trenja se kreću u rasponu od 0,53-0,58;
- hrast-jasen vrednosti kinematskog koeficijenta trenja se kreću u rasponu od 0,58-0,64.

4. ZAKLJUČCI

Dobijeni rezultati merenja statičkog koeficijenta trenja klizanja za različite drvene materijale su u saglasnosti sa literaturnim podacima.

Srednje vrednosti statičkog koeficijenta trenja za kontaktne parove od drveta se kreću u rasponu od 0,42 do 0,63. Statički koeficijent trenja je imao najveće vrednosti za kontaktni par jasen-jasen a najmanje vrednosti za kontaktni par jela-jela.

Veće vrednosti statičkog koeficijenta trenja su dobijene za kontaktne parove od drveta koje karakteriše veća tvrdoća i veći modul elastičnosti, i obrnuto.

Trenje u nekim slučajevima može biti negativna, a u nekim slučajevima i pozitivna pojava. Stoga, na osnovu eksperimetalni istraživanja, se može zaključiti sledeće:

- ako je funkcija cilja optimizacije tribomehaničkog sistema minimum trenja kao elemente kontaktnih parova od drveta treba koristiti jelu, bor ili njihovu kombinaciju;
- ako je funkcija cilja optimizacije tribomehaničkog sistema maksimum trenja kao elemente kontaktnih parova od drveta treba koristiti hrast, jasen ili njihovu kombinaciju.

Buduća istraživanja biće usmerena na merenje statičkog koeficijenta trenja kontaktnih elemenata napravljenih od drugih vrsta drveta, sa različitim mehaničkim i fizičkim karakteristikama i pod različitim eksperimentalnim uslovima.

5. LITERATURA

- [1] B. Luo, L. Li, M. Xu, H. Liu, F. Xing: Analysis of Static Friction Coefficient Between Work-piece and Rubber Belt in Sanding Wood-Based Panel, *Bioresources*, Vol. 9, No. 4, pp. 7372-7381, 2014.
- [2] W. Yin, Z. Liu, P. Tian, D. Tao, Y. Meng, Z. Han, Y. Tian: Tribological properties of wood as a cellular fiber - reinforced composite, *Biotribology*, Vol. 5, pp. 67-73, 2016.
- [3] P. Beer: In situ examinations of the friction properties of chromium coated tools in contact with wet wood, *Tribology Letters*, Vol. 18, No. 3, pp. 373-376, 2005.
- [4] R. Bendikiene, G. Keturakis: The influence of technical characteristics of wood milling tools on its wear performance, *Journal of Wood Science*, Vol. 63, No. 6, pp. 606-614, 2017.
- [5] I.M. Hutchings: Leonardo da Vinci's studies of friction, *Wear*, Vol. 360-361, pp. 51-66, 2016.
- [6] P.J. Blau: The significance and use of the friction coefficient, *Tribology International*, Vol. 34, No. 9, pp. 585-591, 2001.
- [7] V. S. Glass, L. S. Zelinka: Moisture Relations and Physical Properties of Wood, in R. J. Ross (Ed.), *Wood Handbook: Wood as an Engineering Material*, Madison, Wisconsin, pp. 1-19, 2010.
- [8] A.A. Pitenis, D. Dowson, G.W. Sawyer: Leonardo da Vinci's Friction Experiments: An Old Story Acknowledged and Repeated, *Tribology Letters*, Vol. 56, No. 3, pp. 509-515, 2014.
- [9] J.R. Aira, F. Arriaga, G. Iniguez-Gonzalez, J. Crespo: Static and kinetic friction coefficients of Scots pine (*Pinus sylvestris* L.), parallel and perpendicular to grain direction, *Materiales de Construcción*, Vol. 64, No. 315, pp. e030, 2014.
- [10] M. Xu, L. Li, M. Wang, B. Luo: Effects of Surface Roughness and Wood Grain on the Friction Coefficient of Wooden Materials for Wood-Wood Frictional Pair, *Tribology Transactions*, Vol. 57, No. 5, pp. 871-878, 2014.
- [11] R. Sathre, T. Gorman: Improving the performance of wooden journal, *Forest Products Journal*, Vol. 55, No. 11, pp. 41-47, 2005.
- [12] D. Vukelic, Z. Santosi, M. Sokac, I. Budak, T. Saric, G. Simunovic, B. Tadic: Evaluation of the kinetic friction coefficient by using „disc-block“ friction pair of different wooden samples, 16. International Conference on Tribology, Kragujevac: Faculty of Engineering, 15-17 May, 2019, pp. 287-292.

Kratka biografija:



Nikola Đokić rođen je u Vrbasu 1990. god. Master rad na Fakultetu tehničkih nauka iz oblasti Mašinskog inženjerstva odbranio je 2019.god.
kontakt: nikola.dj90@yahoo.com



Đorđe Vukelić rođen je u Novom Sadu 1974. Doktorirao je na Fakultetu tehničkih nauka 2010. Od 2015. je zvanju vanrednog profesora.

ISTRAŽIVANJE STRUJANJA KROZ CEV MEŠAVINE VODE I ETILEN GLIKOLA SA NANOČESTICAMA SREBRA

RESEARCH OF THE FLOW OF WATER/ETHYLENE – GLYCOL MIXTURE WITH SILVER NANOPARTICLES TROUGH PIPE

Nikola Vranješević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj – Predmet ovog rada su hidrauličke karakteristike strujanja mešavine vode i etilen glikola sa nanočesticama srebra kroz pravu cev. Cilj ovog rada je da se utvrde hidrauličke karakteristike strujanja razmatranog nanofluida kroz pravu cev pri laminarnom režimu i stacionarnim uslovima strujanja. Istraživanje je urađeno primenom računarske dinamike fluida (computational fluid dynamics – CFD). Utvrđeno je da pad pritiska raste sa porastom Reynoldsovog broja i koncentracijom nanočestica srebra razmatranog nanofluida.

Abstract – The subject of this paper is the hydraulics characteristics of the flow of water/ethylene – glycol mixture with silver nanoparicles trough the straight pipe. Purpose of this paper is to determine the hydraulics characteristics of the flow of the considered nanofluid through the straight pipe in the laminar regime and the steady flow conditions. The research was done by using the computational fluid dynamics – CFD. It was found that the pressure drop increased with the increase of the Reynolds number and the concentration of silver nanoparticles in the considered nanofluid.

Ključne reči: Računarska dinamika fluida, nanofluidi, pad pritiska, koeficijent trenja.

1. UVOD

U radu će biti ispitane hidrauličke karakteristike nanofluida primenom računarske dinamike fluida. Koristiće se komercijalni softverski program Star CCM+. Hidrauličke karakteristike razmatranog nanofluida bile su predmet eksperimentalnog istraživanja u prošlosti. U radu će se ponoviti sprovedeno eksperimentalno istraživanje predmetnog nanofluida samo sada primenom računarske dinamike fluida. Koristiće se identična geometrija fizičkog modela u softveru bez aproksimacija. Numeričke simulacije podesiće se na osnovu fizičkih svojstava dobijenih predhodnim eksperimentalnim istraživanjem. Zid cevi će se posmatrati kao adijabatna površina, što je odstupanje od fizičkog modela. Kod fizičkog modela toplota je odvođena preko spoljašnje površine cevi. Numeričke simulacije će se izvesti za šest različitih zapreminskih koncentracija nanočestica u osnovnom fluidu (0%, 0,05%, 0,1%, 0,15%, 0,3% i 0,45%).

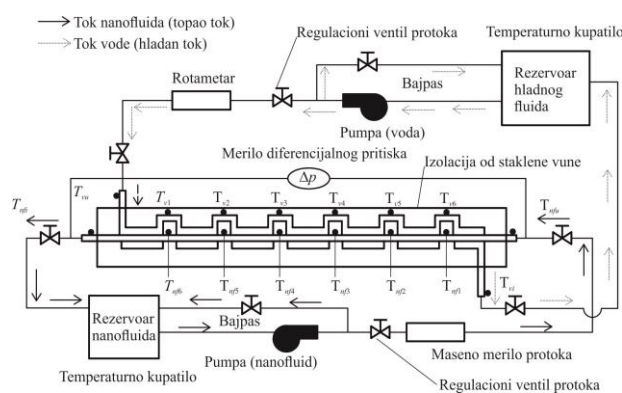
NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Siniša Bikić.

Temperatura nanofluida na ulazu u cev biće 10 °C. Za svaki navedeni slučaj, maseni protok nanofluida će se varirati od 5 g/s do 30 g/s. Na ovaj način će se numeričke simulacije predmetnog fluida uraditi za laminaran režim i stacionarne uslove strujanja. Verifikacija podataka numeričke simulacije uradiće sa vodom primenom Darsi – Vajbahovog (Darcy – Weibach) modela.

2. OPIS EKSPERIMENTALNOG POSTROJENJA

Na slici 3.1 prikazano je eksperimentalno postrojenje za ispitivanje nanofluida korišćeno u predhodnom istraživanju [1]. Eksperimentalno ispitivanje hidrauličkih karakteristika urađeno je na suprotnosmernom razmenjivaču toplote tipa cev u cevi. Dužina unutrašnje cevi razmenjivača iznosi 2,97 m. Unutrašnji prečnici unutrašnje i spoljašnje cevi su 4,3 mm i 10,5 mm.



Slika 3.1 Eksperimentalno postrojenje za ispitivanje nanofluida [1]

Numeričke simulacije urađene su za dva slučaja. U prvom slučaju (slučaj I) nanočestice srebra pomešane su sa surfaktantom u razmeri 50:50. Surfactant je sodijum dodecil sulfat (SDS).

Za ovaj slučaj prethodnim eksperimentalnim istraživanjem su određene jednačine zavisnosti viskoznosti i gustine fluida od temperature i koncentracije čvrstih čestica. U drugom slučaju (slučaj II) nanočestice koje su dodate u osnovni fluid su čisto srebro. Takođe, za određivanje viskoznosti i gustine ovakvog fluida korišćeni su teorijski modeli za fluid sa čvrstim česticama.

3. TEORIJSKE OSNOVE

Prilikom eksperimentalnog merenja viskoznosti Ag/EG-H₂O nanofluida korišćen je Cannon-Fenske prozirni kapilarni viskozimetar. Kapilarnim viskozimetrom merena je kinematska viskoznost nanofluida. Na osnovu dobijene vrednosti kinematske viskoznosti određena je dinamička viskoznost nanofluida iz poznate relacije:

$$\mu = \rho \cdot \nu, \quad (1)$$

gde je ρ – gustina fluida [kg/m³].

Dobijene vrednosti viskoznosti upoređene su sa postojećim modelom za određivanje kinematske viskoznosti. Postojeći model sa kojima je rezultat upoređen je Anštajnov (*Einstein*) model:

$$\frac{\mu_{nf}}{\mu_{of}} = (1 - \phi)^{-2.5}, \quad (2)$$

gde su:

μ_{nf} – dinamička viskoznosti nanofluida [Pa·s];

μ_{of} – dinamička viskoznosti osnovnog fluida [Pa·s];

ϕ – zapreminski udeo nanočestica [-].

Upoređivanjem rezultata eksperimenta i teorijskog modela predložena je jednačina koja uzima u obzir i temperaturu i zapreminski udeo nanočestica [1]:

$$\frac{\mu_{nf}}{\mu_{of}} = 1,0511 - 0,9641\phi + 0,439\phi T - 0,0004T, \quad (3)$$

gde su:

μ_{nf} – dinamička viskoznosti nanofluida [Pa·s];

μ_{of} – dinamička viskoznost osnovnog fluida [Pa·s];

ϕ – zapreminski udeo nanočestica [-] i

T – temperatura nanofluida [K].

Merenje gustine Ag/EG-H₂O nanofluida je urađeno pomoću elektronske vage i menzure zapremine 20 mL. Gustina nanofluida određena je pomoću sledeće jednačine:

$$\rho_{nf} = \frac{m_t - m_{prm}}{V_{nf}}, \quad (4)$$

gde su:

m_t – ukupna masa nanofluida i menzure [kg];

m_{prm} – masa prazne menzure [kg] i

V_{nf} – zapremina nanofluida [m³].

Izmerena gustina nanofluida upoređena je sa postojećim modelom Paka i Čoa (*Pak and Cho*):

$$\rho_{nf} = (1 - \phi)\rho_{of} + \phi \cdot \rho_p, \quad (5)$$

gde su:

ρ_{of} – gustina osnovnog fluida [kg/m³];

ρ_p – gustina materijala nanočestice [kg/m³] i

ϕ – zapreminski udeo nanočestica [-].

Upoređivanjem eksperimentalnih podataka sa postojećim modelom dobijena je jednačina za određivanje gustine nanofluida koja uzima u obzir i temperaturu i zapreminski udeo nanočestica:

$$\frac{\rho_{nf}}{\rho_{of}} = 1 + 0,041\phi + 0,0012\phi T + 4,276 \times 10^{-5} T, \quad (6)$$

gde je T – temperatura nanofluida [K].

Tokom eksperimenta podešavan je maseni protok. Poznavanjem masenog protoka iz izraza za određivanje masenog protoka određena je brzina strujanja:

$$\nu = \frac{4 \cdot \dot{m}}{D^2 \cdot \pi \cdot \rho}, \quad (7)$$

gde su:

$\dot{m}$ – maseni protok [kg/s];

ν – brzina fluida [m/s];

ρ – gustina fluida [kg/m³] i

D – unutrašnji prečnik cevi [m].

Poznavanjem prethodno navedenih veličina određen je Reynoldsov broj iz dobro poznate relacije:

$$Re = \frac{\nu \cdot D}{\nu}, \quad (8)$$

gde su:

Re – Reynoldsov broj [-] i

ν – kinematska viskoznost fluida [m²/s].

Eksperiment je izveden za laminarno strujanje fluida. Koeficijent trenja za laminaran režim strujanja određen je iz sledeće relacije:

$$\lambda = \frac{64}{Re}, \quad (9)$$

gde je λ – koeficijent trenja [-].

Pad pritiska na ravnoj, pravoj deonici definisan je Darsi – Vajbahovim (*Darcy – Weibach*) modelom:

$$\Delta p = \rho \cdot \lambda \cdot \frac{L}{D} \cdot \frac{\nu^2}{2}, \quad (10)$$

gde su:

Δp – pad pritiska [Pa] i

L – dužina cevi [m].

4. REZULTATI I DISKUSIJA

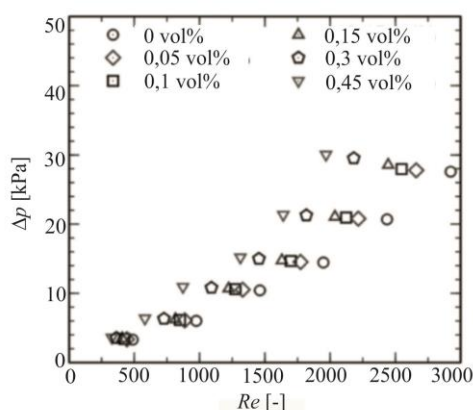
Iz literature [2] su preuzete vrednosti za gustinu i viskoznost vode i pomoću Darsi – Vajbahovog (*Darcy – Weibach*) modela izračunat je pad pritiska za geometriju modela. Dobijeni rezultati su upoređeni sa odgovarajućim rezultatima dobijenim numeričkim simulacijama. U tabeli 1 prikazani su rezultati verifikacije numeričke simulacije vodom primenom Darsi Vajbahovog modela.

Tabela 1 Rezultati verifikacije numeričke simulacije

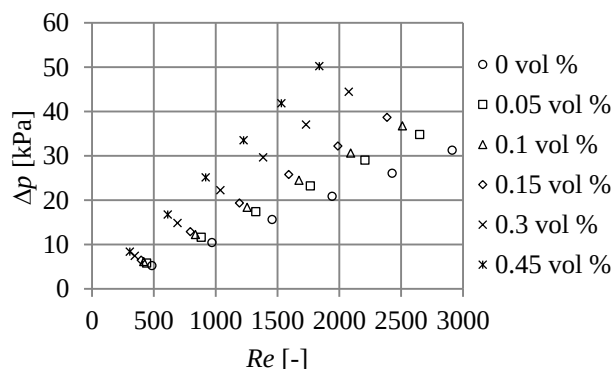
T (°C)	2		5		10	
$\dot{m}$ (g/s)	5	10	5	10	5	10
Re (-)	875	1750	957	1914	1133	2267
$\Delta p_{teo.}$ (Pa)	2994	5988	2737	5475	2312	4623
$\Delta p_{izm.}$ (Pa)	3072	6295	2816	5781	2390	4929
ε (%)	2,6	5,1	2,8	5,6	3,4	6,6

Zavisnost pada pritiska od Reynoldsovog broja i koncentracije nanočestica za temperaturu od 10 °C dobijena eksperimentalnim putem prikazana je na slici 2.

Zavisnost pada pritiska od Reynoldsovog broja i koncentracije nanočestica za temperaturu od 10 °C dobijena računarskom dinamikom fluida za slučaj I prikazana je na slici 3. U oba slučaja pad pritiska ima trend rasta sa povećanjem Reynoldsovog broja i sa povećanjem koncentracije nanočestica u nanofluidu. Kako se koncentracija nanočestica povećava od 0 vol% do 0,15 vol%, povećanje pada pritiska je malo. Iznad 0,15 vol% povećanje pada pritiska je znatno više.

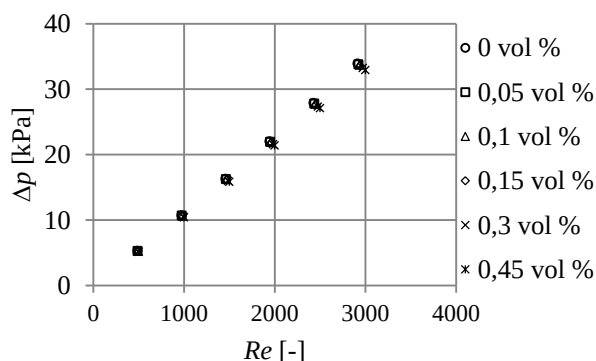


Slika 2 Zavisnost pada pritiska od Reynoldsovog broja i koncentracije nanočestica dobijena eksperimentom [1]



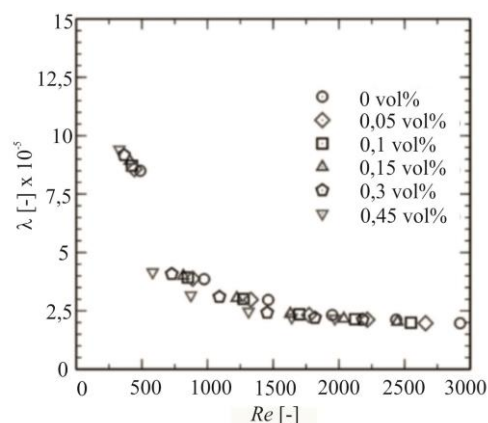
Slika 3 Zavisnost pada pritiska od Reynoldsovog broja i koncentracije nanočestica dobijena računarskom dinamikom fluida – slučaj I

Slika 4 prikazuje rezultate simulacije za slučaj II sa 100% čistim srebrom kao nanočesticama. Na slici se vidi da se vrednost pada pritiska menja isključivo sa Reynoldsovim brojem.



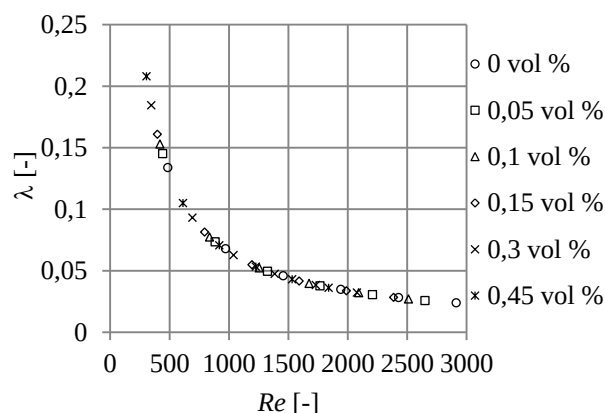
Slika 4 Zavisnost pada pritiska od Reynoldsovog broja i koncentracije nanočestica dobijena računarskom dinamikom fluida – slučaj II

Razlog za tako nešto leži u matematičkom modelu za računanje viskoznosti. (2) koji je ovde upotrebljen, a koji ne uzima u obzor uticaj temperature na viskoznost nanofluida. Time se dobijaju jako bliske vrednosti za koeficijent viskoznosti za bilo koju koncentraciju nanočestica, pa se samim tim i rezultati za pad pritiska za različite koncentracije nanočestica poklapaju. Kako bi se postojeći teorijski model mogao upotrebiti u oblasti nanofluida potrebna je njegova korekcija. Na slici 5 prikazana je zavisnost koeficijenta trenja od Reynoldsovog broja i koncentracije nanočestica na temperaturi 10°C dobijena eksperimentalnim putem. Može da se vidi da koeficijent trenja znatno menja sa porastom Reynoldsovog broja, ali da je skoro konstantan ako se posmatra u odnosu na zapreminsku koncentraciju nanočestica. Pri nižim Reynoldsovim brojevima primećuje se uticaj na koeficijent trenja sa promenom zapreminske koncentracije nanočestica. Pri višim Reynoldsovim brojevima povećanje koeficijenta trenja sa promenom koncentracije nanočestica je veoma malo.



Slika 5 Zavisnost koeficijenta trenja od Reynoldsovog broja i koncentracije nanočestica na temperaturi 10 °C dobijena eksperimentalnim putem [1]

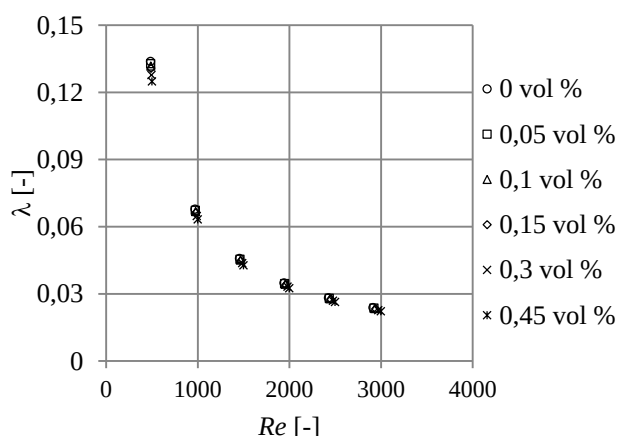
Na slici 6 prikazana je zavisnost koeficijenta trenja od Reynoldsovog broja i koncentracije nanočestica na temperaturi 10°C, dobijena primenom računarske dinamike fluida za slučaj I.



Slika 6 Zavisnost koeficijenta trenja od Reynoldsovog broja i koncentracije nanočestica na temperaturi 10 °C dobijena računarskom dinamikom fluida – slučaj I

Na slici se vidi da se vrednost koeficijenta trenja znatno menja sa promenom Reynoldsovog broja. Kod nižih Reynoldsovih brojeva promena koeficijenta trenja usled promene zapreminske koncentracije nanočestica je znatno izraženija.

Rezultatima numeričke simulacije dobijeni su isti trendovi kao i kod eksperimentalnih podataka. Vrednosti koeficijenta trenja dobijene primenom računarske dinamike fluida i eksperimentalnim putem se u dobroj meri poklapaju. Za slučaj II, *slika 7*, vidi se očigledna zavisnost koeficijenta trenja od Reynoldsovog broja. Kao i u slučaju pada pritiska, u odnosu na zapreminsku koncentraciju nanočestica vrednost koeficijenta trenja gotovo da se ne menja. Razlog su vrednosti viskoznosti koje se dobijaju iz Ajnštajnovne jednačine (2).



Slika 7 Zavisnost koeficijenta trenja od Reynoldsovog broja i koncentracije nanočestica na temperaturi 10 °C dobijena računarskom dinamikom fluida – slučaj II

5. ZAKLJUČAK

Utvrđeno je da hidrauličke karakteristike nanofluida koji je ispitan u ovom radu odgovaraju realnom ponašanju fluida i poklapaju se sa rezultatima eksperimenta. Porast pada pritiska uslovljen je povećanjem Reynoldsovog. Uticaj nanočestica na pad pritiska je takođe značajan, jer nanočestice povećavaju viskoznost fluida.

Reynoldsov broj ima značajan uticaj na koeficijent trenja, dok je uticaj koncentracije nanočestica na koeficijent trenja zanemarljiv. Povećanjem Reynoldsovog broja preko 1000 ulazi se u zonu kada koeficijent trenja teži da bude konstantan.

Predložene jednačine za određivanje gustine i viskoznosti dobijene eksperimentalnim putem za nanofluid sa nanočesticama srebra su na ovaj način potvrđene primenom računarske dinamike fluida. Takođe je potvrđeno da je teorijski model za određivanje viskoznosti fluida sa dodatim česticama nepouzdan za razmatrani nanofluid jer ne obuhvata sve faktore koji imaju uticaj na ponašanje fluida.

6. REFERENCE

- [1] C. Selvam, D. Mohan Lal and Sivasankaran Harish, *Convective heat transfer behaviour of water-ethylene glycol-mixture with silver nanoparticles under laminar flow conditions*, Journal of Mechanical Science and Technology 32 (5) (2018) 2191~2199.
- [2] J. V. Sengers, J. T. R. Watson: *Improved international formulations for the viscosity and thermal conductivity of water substance*, J. Phys.Chem.Ref. Data, 15:1291, (1986).

Kratka biografija:



Nikola Vranješević, rođen je u Vršcu 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Mašinstva – Naftna i gasna tehnika odbranio je 2019. god.

**ANALIZA TRANZIJENTNIH KARAKTERISTIKA BOČNE DINAMIKE VOZILA
POMOĆU SOFTVERA ZA MODELIRANJE I SIMULACIJE****ANALYSIS OF TRANSIENT CHARACTERISTICS OF LATERAL VEHICLE DYNAMICS
WITH HELP OF SOFTWARE FOR MODELING AND SIMULATIONS***Dalibor Pelagić, Fakultet tehničkih nauka, Novi Sad***Oblast – MAŠINSTVO**

Kratak sadržaj – U ovom radu je izvršena analiza uticajnih parametara na ponašanje motornih vozila pri naglim promjenama pravca kretanja, na primjeru izolovanih, pojednostavljenih manevara. Kao glavni alati za pomenutu analizu korišteni su softver Adams Car i analitički proračun uz pomoć softvera Microsoft Excel.

Ključne reči: skretanje vozila, tranzijentni odziv, Adams Car

Abstract – This paper represents analysis of influential parameters on vehicle behavior during sharp cornering, on example of isolated and simplified maneuvers. The main tools used for this analysis are softwer Adams Car and analytic calculation with help of softwer Microsoft Excel.

Keywords: vehicle cornering, transient response, Adams Car

1. UVOD

Ponašanje motornih vozila pri savladavanju krivina određenih radijusa zavisi od velikog broja različitih parametara. Da bi se skretanje pravilno izvelo, sa što manje rizika za gubljenje kontrole, potrebna je precizna izrada pojedinih podsistema vozila, kao i pravilno upravljanje vozilom.

Npr., osnova od koje se polazi pri analiziranju ponašanja vozila u ovakvim situacijama, jeste geometrija i ponašanje sistema upravljanja i sistema vođenja točkova. Pored toga, jako je bitna kinematika pojedinih elemenata, tokom izvođenja određenih manevara. Dinamika vozila i njegovih podsistema, kao i njihovo ponašanje pri određenim pobudama, je, takođe, aspekt od krucijalnog značaja. Na kraju, „komunikacija“ između vozila i rukovaoca vozilom veoma je važna za sigurno održavanje kontrole nad vozilom.

Cilj rada jeste analiza uticajnih parametara na ponašanje vozila pri naglim promjenama pravca kretanja. Uz pomoć CAE simulacija na kompleksnom modelu i analitičkog proračuna za uprošćeni model dobijeni su određeni rezultati. Za tačno određene ulazne parametre dobijeni su određeni izlazni parametri, nakon čega je izvršena njihova analiza.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Boris Stojić.

2. TEORIJSKE OSNOVE BOČNE DINAMIKE VOZILA

Ponašanje motornog vozila pri promjeni pravca kretanja je važna osobina vozila, za koju se najčešće koristi izraz upravljivost [1]. Upravlјivost je izraz, kojim se označava odziv vozila na određenu pobudu, koju izaziva rukovalac vozila (vozača). Može se, takođe, reći da upravljivost opisuje vezu između vozača i vozila. Vozač i vozilo predstavljaju jedan „zatvoren sistem“, u smislu da vozač daje komandu za način i pravac kretanja vozila, a takođe, isto tako, po potrebi, vrši korekciju zadanih komandi, kako bi postigao željeni način i pravac kretanja, ili željenu poziciju vozila, dok vozilo vraća povratnu informaciju u vidu odziva na zadatu komandu. U cilju određivanja ponašanja samog vozila, koristi se tzv. „otvorena petlja“ [1]. Otvorena petlja se koristi kako bi se iskazao odziv vozila na tačno određeni način aktiviranja vozačkih komandi.

2.1. Stacionarno stanje

Stacionarno stanje podrazumijeva kretanje vozila, pri kojem se parametri kretanja ne mijenjaju ili teže da se vrate na početnu vrijednost, ukoliko iz nekog razloga dođe do promjene. Stacionarno kretanje može biti translatorno, a može biti i kretanja po kružnici, odnosno obrtanje krutog tijela oko nepokretne ose, što odgovara kretanju vozila kroz krivinu određenog radijusa. Tokom ovakvog kretanja, vrijednosti ugla upravljača i brzine vozila su konstantne. Ono što se najčešće mjeri u otvorenoj petlji, jeste gradijent upravljivosti. Gradijent upravljivosti je veličina, koja određuje kako se potreban ugao zakretanja volana, za isti radiju krivine, mijenja sa promjenom bočnog ubrzanja vozila.

2.1.1. Skretanje pri malim brzinama

Tokom skretanja vozila pri malim brzinama (npr. pri parkiranju vozila), usljed veoma malih bočnih ubrzanja, bočne sile na pneumaticima, kao i vrijednosti povodenja točkova, su zanemarlјivo male. Prema tome, u takvim uslovima kretanja, pneumatici se kotrlјaju bez proklizavanja. Sve tačke vozila se kreću po koncentričnim kružnicama, čiji se radijusi i položaji centra ne mijenjaju u vremenu. Ovo predstavlјa specijalan slučaj kretanja i analiza ovakvog kretanja služi za određivanje osnovnih karakteristika upravljivosti i stabilnosti.

Ako nema povodenja zadnjih točkova, centar zakretanja vozila mora ležati na pravoj, koja prolazi kroz zadnju osovinu. Poluprečnici skretanja oba prednja točka moraju da potiču iz istog centra zakretanja. Ukoliko se centri skretanja prednjih točkova ne bi poklapali, došlo bi do

stvaranja neželjenih sila na prednjim pneumaticima tokom skretanja, do određenog proklizavanja i do dodatnog i ubzanog trošenja prednjih pneumatica.

Srednji ugao zakretanja prednjih točkova naziva se Akermanovim uglom [1] i izražava se kao:

$$\delta = \frac{L}{R_C} \quad (1)$$

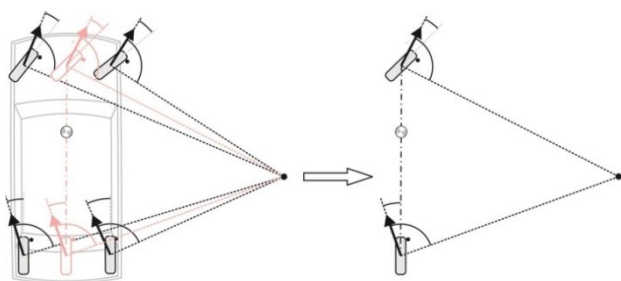
gdje je:

L - osovinsko rastojanje

R_C - radijus krivine putanje centra mase vozila

Pojam „Akermanov ugao“ ili „Akermanova geometrija“ se koristi pri označavanju pravilne geometrije prednjih točkova pri skretanju. Tačne vrijednosti uglova zakretanja prednjih točkova zavise od osovinskog rastojanja i radijusa skretanja vozila

Radi lakše analize, uvodi se tzv. model vozila sa jednim tragom („bicikl model“), prikazan na slici 1.



Slika 1. Model vozila sa jednim tragom [2]

Ovakav model je baziran na sljedećim uprošćenjima:

- Visina centra mase vozila u odnosu na podlogu je nula, odnosno centar mase leži na površini podloge. Ovim se eliminiše razlika između vrijednosti bočnih sila na spoljašnjim i unutrašnjim točkovima. Usljed toga, dva točka (desni i lijevi) se mogu svesti na jedan.
- Izrazi koji opisuju kretanje su linearizovani. Linearizacija važi za uglove manje od 4°. Ova linearizacija takođe obuhvata i linearizaciju ponašanja pneumatica.

2.1.2. Skretanje pri velikim brzinama

Razlika u uslovima skretanja pri većim brzinama (kojima se vozilo kreće u saobraćaju) u odnosu na male brzine, jeste pojava izraženijeg bočnog ubrzanja pri većim brzinama. Da bi se bočne sile, koje djeluju na vozilo uravnotežile, na točkovima se razvijaju bočne sile suprotnog smjera. Takođe, u ovakvim uslovima, na svim točkovima će biti prisutna određena bočna povođenja.

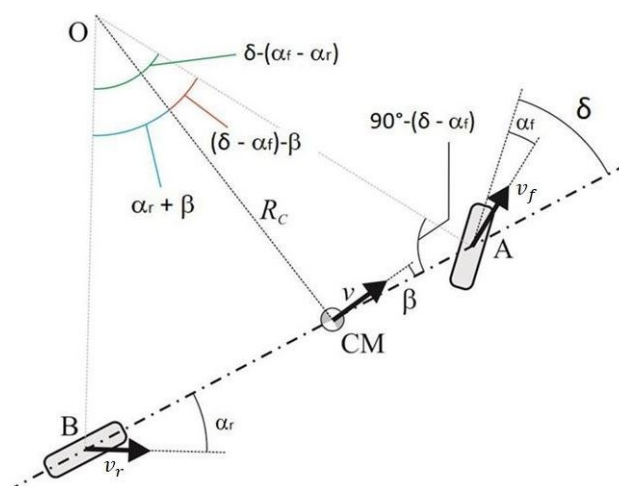
Analizom geometrije sa slike 2, može se napisati izraz:

$$\delta = \frac{L}{R_C} + \alpha_f - \alpha_r \quad (1)$$

gdje je:

α_f - ugao povođenja prednjih točkova

α_r - ugao povođenja zadnjih točkova



Slika 2. Analiza geometrije skretanja vozila pri većim brzinama [2]

Uz pomoć dinamičke analize i uvođenjem dodatnih pojednostavljenja [2]:

- uglovi povođenja točkova su mali
- radijus krivine putanje vozila je veliki
- radijus krivine je konstantan
- srednji ugao zakrenutosti prednjih točkova je konstantan,

dobija se izraz:

$$\delta = \frac{L}{R_C} + \left(\frac{G_f}{C_{af}} - \frac{G_r}{C_{ar}} \right) \cdot \frac{v^2}{g \cdot R_C} \quad (2)$$

gdje je:

C_{ar} - bočna krutost zadnjih pneumatica

C_{af} - bočna krut. prednjih pneumatica

G_r - stat. opt. zadnje osov.

G_f - stat. opt. prednje osov.

v - transl. brzina vozila

g - ubrzanje zemljine teže

Uzimanjem u obzir izraza za gradijent upravljivosti, dobija se:

$$\delta = \frac{L}{R_C} + K \cdot \frac{a_y}{g} \quad (3)$$

gdje je:

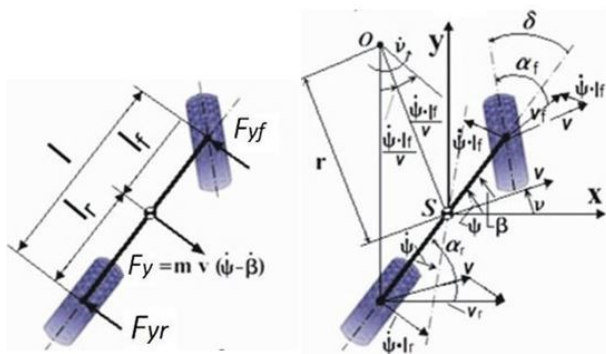
K - gradijent upravljivosti

a_y - bočno ubrzanje vozila

Od gradijenta upravljivosti zavise potrebni ulazni parametri za sistem upravljanja, pri savladavanju krivine određenog radijusa R . Postoje tri različita karaktera upravljivosti, pomoću kojih se opisuje ponašanje vozila: neutralna upravljivost, podupravljivost i nadupravljivost.

2.1. Nestacionarno stanje

Nestacionarno kretanje vozila podrazumijeva kretanje u toku kojeg se mijenjaju vrijednosti parametara kretanja vozila (translatorna brzina kretanja, ugaona brzina, bočno ubrzanje...)



Slika 3. Analiza dinamike vozila tokom nestacionarnog kretanja [3]

Nakon dinamičke analize sa slike 3, uz pretpostavku da je brzina vozila konstantna, dobija se izraz:

$$\ddot{\beta} + \left[\frac{C_{af} + C_{ar}}{m \cdot v} + \frac{C_{af} \cdot l_f^2 + C_{ar} \cdot l_r^2}{\theta \cdot v} \right] \cdot \dot{\beta} + \left[\frac{C_{ar} \cdot l_r - C_{af} \cdot l_f}{\theta} + \frac{C_{af} \cdot C_{ar} \cdot L^2}{\theta \cdot m \cdot v^2} \right] \cdot \beta = \left[-C_{af} \cdot \theta \right] \cdot \dot{\delta} + \left[\frac{C_{af} \cdot l_f}{\theta} - \frac{C_{af} \cdot C_{ar} \cdot l_r \cdot L}{\theta \cdot m \cdot v^2} \right] \cdot \delta \quad [3] \quad (4)$$

gdje je:

β - ugao plivanja c.m. l_r - rast. c.m. od zadnje os.
 $\dot{\beta}$ - prvi izvod β po vrem. L - osovinsko rastojanje
 $\ddot{\beta}$ - drugi izvod β po vrem. θ - mom. inert. oko z ose
 m - masa vozila δ - ugao zakrenutosti točk.
 l_f - rast. c.m. od prednje os. $\dot{\delta}$ - prvi izvod δ po vrem.

Dio prethodne jednačine, desno od znaka jednakosti, predstavlja homogeni dio jednačine i može se definisati kao smetnje u toku kretanja, jer realne smetnje (neravnomjernost podloge i aerodinamičke smetnje) nisu uzete u obzir.

Lijevo od znaka jednakosti je homogeni dio jednačine. Može se primijetiti da ovaj dio ima isti oblik kao uobičajeni izraz za prigušenje vibracija:

$$\ddot{\beta} + 2 \cdot D \cdot \omega_e \cdot \dot{\beta} + \omega_e^2 \cdot \beta = 0 \quad [3] \quad (5)$$

Na osnovu izraza (4) i (5) moguće je odrediti sljedeće parametre prigušenja:

➤ Neprigušena spostvena frekvencija:

$$\omega_e = \sqrt{\frac{C_{ar} \cdot l_r - C_{af} \cdot l_f}{\theta} + \frac{C_{af} \cdot C_{ar} \cdot L^2}{\theta \cdot m \cdot v^2}} \quad (6)$$

➤ Koeficijent prigušenja:

$$D = \frac{1}{2 \cdot \omega_e} \cdot \left[\frac{C_{af} + C_{ar}}{m \cdot v} + \frac{C_{af} \cdot l_f^2 + C_{ar} \cdot l_r^2}{\theta \cdot v} \right] \quad (7)$$

➤ Prigušena sopstvena frekvencija:

$$\omega_{ed} = \omega_e \cdot \sqrt{1 - D^2} \quad (8)$$

3. OPIS SOFTVERA ZA SIMULACIJU I MODELIRANJE „ADAMS CAR“

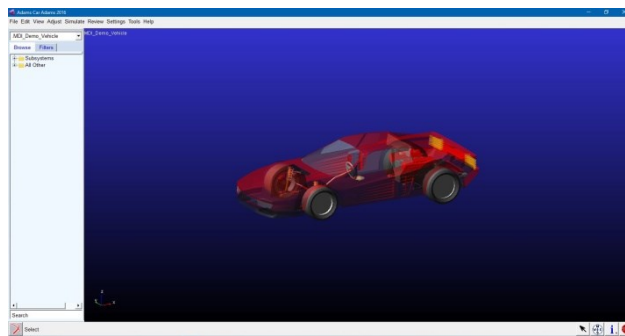
3.1. Pristup dinamici uz pomoć softvera „Adams“

Ovo je softver koji pomaže pri proučavanju dinamike pokretnih elemenata i analiziranju uticaja i rasporeda opterećenja i sila, koji djeluju na mehaničke sisteme. Zajedno sa širokim spektrom mogućih analiza optimizovan je za rješavanje najkompleksnijih dinamičkih problema.

Kao najpoznatiji softver za proučavanje dinamike mehaničkih sistema, Adams doprinosi poboljšavanju efikasnosti inženjera i smanjuje troškove dobijanja konačnog proizvoda, omogućavajući njegovu raniju validaciju kao dinamički uravnoteženog sistema.

3.2. Opis dopunskog modula „Adams Car“

Adams Car je dio softvera Adams, koji služi modeliranju i ispitivanju vozila. Glavna svrha korištenja ovog softvera u automobilskoj industriji, jeste simulacija ponašanja modela vozila, kao i njegovih podsistema pojedinačno. Ovi modeli se npr. često koriste za simulaciju ponašanja vozila pri dejstvu nelinearnih vibracija velikih amplituda, kao što su vibracije koje potiču od pogonskog agregata ili transmisije.



Slika 4. Model vozila u softverskom okruženju

3.2.1. Manevri u otvorenoj petlji

Sa aspekta kontrole izlaznih parametara nekog sistema, postoji sistem sa kontrolom u otvorenoj i sistem sa kontrolom u zatvorenoj petlji. Suštinska razlika između ovih sistema je u tome što sistem sa otvorenom petljom ne daje povratnu informaciju na ulaz u sistem.

Adams Car, pored ostalog, nudi mogućnost simulacije različitih manevara u otvorenoj petlji: Drift, Fish-Hook, Impulse steer, Ramp steer, Single Lane-Change, Step steer, Swept-Sine steer.

Manevar promjene ugla upravljača po odskočnoj funkciji (Step steer) se koristi za analizu izlaznih parametara u vremenskom domenu i korišten je u simulaciji predstavljenoj u glavnom dijelu rada.

4. ANALIZA PRORAČUNSKIH I SIMULACIJSKIH REZULTATA

4.1. Proračun parametara tranzijentnog odziva vozila na bazi pojednostavljenog modela

U ovom potpoglavlju su analitičkim putem određeni parametri tranzijentnog odziva vozila, usljed impulsne pobude. Ova pobuda se manifestuje kratkotrajnim dejstvom bočne sile na vozilo, koje izaziva kratkotrajni izlazak vozila iz stacionarnog stanja.

Korištenjem parametara vozila, usvojenim iz baze podataka Adams Car-a, i izraza (6), (7) i (8), za tri različite brzine dobijeni su sljedeći rezultati:

Pri 40 km/h:

$$\omega_e = 29.49 \text{ Hz}, D = 1.03, \omega_{ed} = 29.49 \cdot \sqrt{-0.0609} \text{ Hz}$$

Pri 80 km/h:

$$\omega_e = 16.16 \text{ Hz}, D = 0.94, \omega_{ed} = 0.33 \text{ Hz}$$

Pri 120 km/h:

$$\omega_e = 12.18 \text{ Hz}, D = 0.83, \omega_{ed} = 0.55 \text{ Hz}$$

Pri brzini od 40 km/h, odziv vozila je nadkritično prigušen. Pri brzinama od 80 i 120 km/h, i pored smanjenja koeficijenta prigušenja, vrijednost frekvencije sopstvenih prigušenih oscilacija je zanemarljivo mala, tako da se i ove vrijednosti mogu zanemariti i aproksimirati kao nadkritično prigušene.

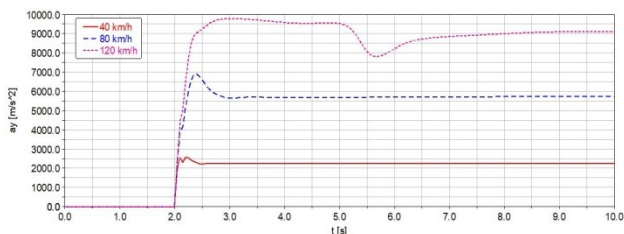
4.2. Manevar promjene ugla upravljača po odskočnoj funkciji

U ovom dijelu rada predstavljena je analiza rezultata, dobijenih simuliranjem manevara promjene ugla upravljača po odskočnoj funkciji, u softverskom okruženju Adams Car. Kao model simulacije korišten je kompleksan model vozila. Osnovni parametri vozila su isti kao kod pojednostavljenog modela, ali je sam model mnogo kompleksniji, a time i mnogo realniji.

Simuliranje je izvedeno u tri iteracije. Brzina je, ponovo, parametar koji ima različite vrijednosti u sve tri iteracije (40, 80 i 120 km/h).

4.2.1. Bočno ubrzanje

U ovom dijelu su prikazane karakteristike bočnog ubrzanja tokom tri navedene iteracije.



Slika 5. Karakteristike bočnog ubrzanja

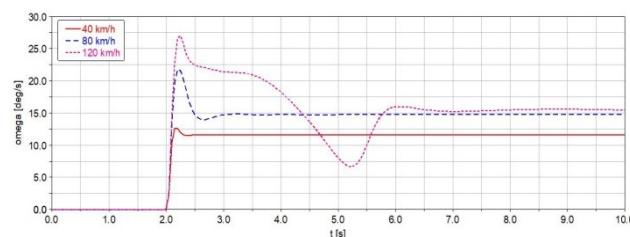
Vrijednost bočnog ubrzanja raste sa porastom brzine kretanja vozila, što je vidljivo na slici 5. Kao što može da se vidi, manevar počinje u drugoj sekundi simulacije i na početku manevara se dešava nagli skok vrijednosti.

U prve dvije iteracije (40 i 80 km/h), vrijednost bočnog ubrzanja blago osciluje i zatim se ustaljuje na nekoj konstantnoj vrijednosti. Maksimum u prvoj iteraciji je $\approx 2.6 \text{ m/s}^2$, dok se u drugoj penje na vrijednost od $\approx 9.8 \text{ m/s}^2$. Ustabiljenje u prvoj iteraciji se dešava pri vrijednosti od $\approx 2.25 \text{ m/s}^2$, a u drugoj pri $\approx 5.75 \text{ m/s}^2$.

Treća iteracija (pri 120 km/h) daje nepravilnu karakteristiku, koja govori da bi realno izvođenje manevara, pri ovoj brzini, bio prilično težak zadatak za rukovaoca vozilom sa prosječnim sposobnostima.

4.2.3. Ugaona brzina

U ovom dijelu su date i analizirane ugaone brzine, kao izlazni parametri simulacija.



Slika 6. Karakteristike ugaonih brzina

Na slici 6 predstavljene su karakteristike ugaone brzine, dobijene u tri navedene iteracije.

Karakter promjene vrijednosti ovog izlaznog parametra je sličan promjeni bočnog ubrzanja. U prve dvije iteracije, vrijednost ugaone brzine skače na neki maksimum, a zatim se uz blage oscilacije ustaljuje na konstantnu vrijednost. Maksimum vrijednosti u prvoj iteraciji je 12.3 °/s, a u drugoj 22 °/s. Ustaljene vrijednosti ugaonih brzina u prvoj i drugoj iteraciji su 11.8 °/s i 15°/s, respektivno.

I u ovom slučaju je vidljiv porast vrijednosti analiziranog izlaznog parametra sa porastom brzine kretanja vozila, što je u skladu sa teorijskim očekivanjima.

Takođe, i kod ovog parametra, karakteristika pri brzini od 120 km/h, ima nepravilan oblik, što potvrđuje pretpostavku o mogućnosti gubitka kontrole nad vozilom, u realnim uslovima izvođenja manevara.

5. ZAKLJUČAK

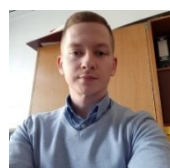
Kod analitički dobijenih rješenja, model je pokazao nadkritična prigušenja. Takva rješenja nisu validna u ocjeni ponašanja vozila. Zaključak je da je model previše pojednostavljen.

Model korišten u softverskoj simulaciji je visoko kompleksan, a time i prilično vjeran realnom modelu. U ovom slučaju, rezultati imaju oscilatorni karakter, što opravdava teorijska očekivanja.

6. LITERATURA

- [1] T. D. Gillespie, „Fundamentals of Vehicle Dynamics”, 1992.
- [2] B. Stojić, „Predavanja iz TKDV“, FTN Novi Sad 2018., *mehanizacija.uns.ac.rs* (pristupljeno u martu 2019.)
- [3] B. Heissig, M. Ersoy, „Chassis Handbook“ 2011.

Kratka biografija:



Dalibor Pelagić rođen je u Brčkom (BiH) 1994. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Mašinstva – Mehanizacija i konstrukciono mašinstvo odbranio je 2017.god., nakon čega je upisao master studije.

SIMULACIONI MODEL METODE ZA GENERISANJE ŠUMA DISKRETNOG U VREMENU I KONTINUALNOG PO VREDNOSTI

SIMULATION MODEL OF CONTINUOUS VALUE NOISE GENERATOR DISCRETE IN TIME

Marko Knežević, Dragan Pejić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu je prikazan simulacioni model nove metode za generisanje naponskog šuma uniformne raspodele vrednosti. Ovaj šum, dither, je ključni element u radu stohastičke analogno digitalne (AD) konverzije. Za razliku od uobičajenog načina generisanja primenom šifra registra sa linearnom povratnom spregom (Linear Feedback Shift Register - LFSR) i digitalno analognog (DA) konvertora nova metoda je zasnovana na asinhronom odabiranju testerastog ili trougaonog napona. Kao osnovni dobitak predložene metode se navodi kontinualnost ditera po vrednosti, za razliku od prethodnog rešenja koje karakteriše diskretan dither. Primena kontinualnog ditera omogućava dalje povećanje efektivne rezolucije stohastičkog AD konvertora.

Ključne reči: AD konvertor, dither u kontinualnom vremenu, generator šuma, programiranje, c++

Abstract – This paper presents simulation model of uniform noise generator of voltage signal. The noise, dither, is the main principle of stochastic AD conversion. Unlike the common way of generation, this method uses Linear Feedback Shift Register – LFSR and it's based on an asynchronous sampling of triangle or sawtooth voltage signal. Main goal of the method is time-continuous jitter which has effect on further increase of effective resolution of stochastic AD converter.

Keywords: AD conversion, time-continuous jitter, noise generator, programming, c++

1. UVOD

Na Katedri za električna merenja Fakulteta tehničkih nauka u Novom Sadu se već 25 godina razvija stohastička AD konverzija [1]. Ova metoda je zasnovana na korišćenju fleš AD konvertora male rezolucije, najčešće rezolucije 2 bita. Mala rezolucija ugrađenog AD konvertora se kompenzuje dodavanjem šuma posebnih svojstava (dither) na ulazni signal, kao i usrednjavanjem velikog broja odbiraka dobijenih na visokoj učestanosti (engleski: *oversampling*). Dither treba da bude šum uniformne funkcije gustine raspodele opsega jednakog +/- polovini kvanta korišćenog fleš AD konvertora. Razvijena teorija [2] predviđa adaptivnost kao vrlo korisnu osobinu stohastičkog konvertora: za duže merenje se dobija manja greška.

NAPOMENA:

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

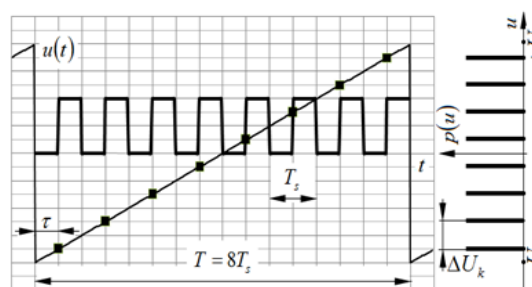
Greška opada sa kvadratnim korenom iz dužine trajanje merenja. Za četiri puta duže merenje, dobija se duplo manja greška. Greška kojom se bavi teorija je isključivo slučajne prirode izazvana prisustvom šuma idealizovanih osobina. U praksi [3] se pokazuje kao ograničavajući faktor prisustvo sistematskih grešaka usled neidealnosti ditera, ali i svih primenjenih komponenti.

U prethodnim godinama razvoja stohastičke metode su otklonjene mnoge sistematske greške [4]. U ovom trenutku se procenjuje da najveća preostala sistematska greška potiče od konačne rezolucije ditterskog signala generisanog primenom DA konvertora.

Predložena je nova metoda [5] koja daje mogućnost generisanja ditera bez primene DA konvertora. Ovaj rad se bavi softverskim simulacionim modelom namenjenim za proveru predložene metode.

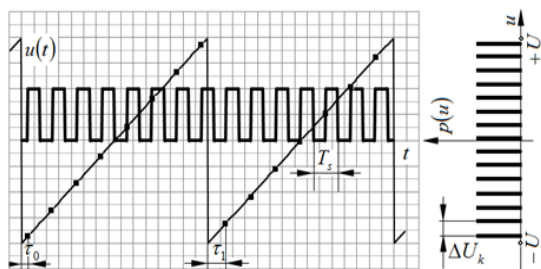
2. PRIKAZ NOVE METODE

Na slici 1. je prikazan testerasti napon $u(t)$, frekvencije $f = \frac{1}{T}$ i impulsi osam puta veće učestanosti $f_s = 8f$. Ako pretpostavimo da se odabiranje testerastog napona vrši na svaku uzlaznu ivicu digitalnog signala, dobićemo 8 vrednosti koje su ekvidistantno raspoređene unutar opsega testerastog napona. Sa desne strane je prikazana diskretna funkcija gustine raspodele vrednosti odbiraka.



Slika 1. Celobrojni odnos učestanosti

Na slici 2. su prikazane dve periode testerastog napona i impulsi učestanosti $f_s = \left(8 - \frac{1}{2}\right)f = 7.5f$. Sada će se, u dve susedne periode testerastog napona, desiti 15 ekvidistantno raspoređenih vrednosti odbiraka. Na ovaj način se može vršiti dalje uopštavanje, čiji rezultat je sve veći broj mogućih vrednosti odbiraka. Bitno je naglasiti da se pri ovome ne povećava učestanost odabiranja, što je jako važno u praktičnoj izvedbi.



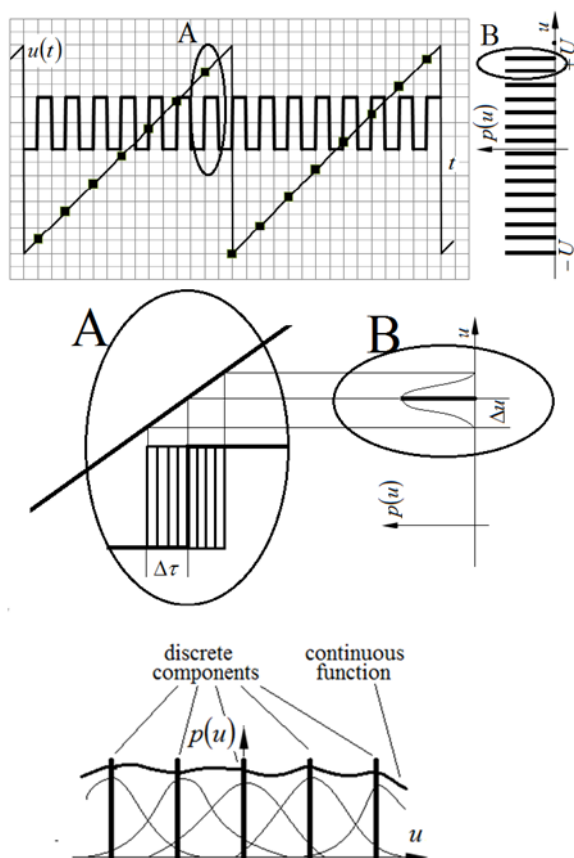
Slika 2. Necelobrojni odnos učestanosti

U Tabeli 1. je prikazan broj mogućih vrednosti odbiraka u zavisnosti od odnosa učestanosti odabiranja i učestanosti testerastog napona.

Tabela 1. Broj mogućih vrednosti

Odnos učestanosti	Broj mogućih vrednosti
$\frac{f_s}{f}$	N
$2^n \pm \frac{1}{2}$	$2^{n+1} \pm 2^0$
$2^n \pm \frac{1}{2} \pm \frac{1}{4}$	$2^{n+2} \pm 2^1 \pm 2^0$
$2^n \pm \frac{1}{2} \pm \frac{1}{4} \pm \frac{1}{8}$	$2^{n+3} \pm 2^2 \pm 2^1 \pm 2^0$

Ako se na prethodnu zamisao doda i mogućnost da se frekvencija odabiranja menja oko vrednosti f_s , umesto diskretne se dobija kontinualna funkcija gustine raspodele. Na slici 3. je ilustrovana takva situacija.



Slika 3. Uticaj ditera na gustinu raspodele

Usled stalnog oscilovanja učestanosti f_s , dolazi do oscilovanja položaja prednje ivice koja definiše trenutak odabiranja, pa će se odabiranje desiti malo ranije ili malo kasnije. Posledica ovoga je dobijanje nešto manje ili nešto veće vrednosti odbirka u poređenju sa situacijom kada nema oscilovanja. Umesto jedne diskretne vrednosti odbirka, sada dobijamo kontinualan skup mogućih vrednosti. Na osnovu [6] sledi da će suma mnoštva slučajnih promenljivih čije su srednje vrednosti ekvidistantno raspoređene težiti uniformnoj raspodeli (Gaus Step Method).

Osnovna zamisao nove metode za generisanje ditera kontinualnog skupa vrednosti se sastoji od odabiranja jednom u okviru periode testerastog signala. Trenutak odabiranja je određen: a) pseudoslučajnom vrednošću iz LFSR, b) odnosom učestanosti $\frac{f_s}{f}$ i c) indeksom modulacije učestanosti f_s .

3. SIMULACIONI MODEL

Simulacioni model je realizovan u C++. Korisnik može zadati parametre koji određuju način odabiranja testerastog napona.

3.1. Struktura LFSR

LFSR je definisan:

- Dužinom šift registra. Predviđeno je da će maksimalna dužina biti 32 bita, tako da se šift registar simulira promenljivom unsigned integer tipa.
- Pozicijama sa kojih se uzima povratna sprega. U [7] je data tabela sa optimalnim povratnim spregama koje obezbeđuju maksimalno dugačku pseudoslučajnu sekvencu.
- Vrsta povratne sprege: preko XNOR ili XOR kola.
- Pozicije sa kojih se uzimaju bitovi i formira vrednost koja definiše trenutak odabiranja u okviru jedne periode testerastog napona. Za m definisanih pozicija, srednja vrednost odnosa učestanosti treba da bude 2^m .

3.2. Oblik ulaznog napona

Teorija se bavi testerastim naponom, a u praksi se koristi trougaoni napon, pošto je vrlo teško izvodivo realizovati beskonačno strmu silaznu ivicu testerastog napona. Oba napona imaju uniformnu funkciju gustine raspodele vrednosti, pa se očekuju isti rezultati korišćenjem oba talasna oblika.

3.3. Način modulacije

Predviđena su dva načina modulacije učestanosti odabiranja: deterministički i stohastički. Kod determinističkog načina modulacije realizovana je promena učestanosti f_s po trougaonoj funkciji. Kod stohastičke modulacije f_s je odabrana trougaona funkcija gustine raspodele odstupanja frekvencije f_s .

3.4 Odnos učestanosti

Odnos učestanosti se zadaje u obliku $N, 1, -1, 1 \dots$ pri čemu je $N = 2^m$ a ostali brojevi predstavljaju koeficijente polinoma $(\frac{1}{2}, \frac{1}{4}, \frac{1}{8} \dots)$. Pored celobrojnog odnosa učestanosti, moguće je definisati i necelobrojan odnos učestanosti, takav da ga je moguće lako realizovati hardverom na bazi fazno zaključane petlje i dva delitelja učestanosti [5].

3.5. Indeks modulacije

Bilo da je odabran deterministički ili stohastički način variranja učestanosti odabiranja, indeks modulacije predstavlja amplitudu promene učestanosti relativno iskazanu u odnosu na učestanost f_s .

3.6 Način ispisa u fajl

Predviđena su dva načina zapisa. U prvom slučaju se u fajl zapisuju realni brojevi kao vrednosti odbiraka. Ovako organizovan fajl je namenjen za statističke testove koji proveravaju uniformnost i druge stohastičke osobine ditera. Drugi način zapisa je u obliku niza nula i jedinica dobijenih iz binarne reprezentacije odbiraka napona. Ovakav fajl je namenjen za statističke testove koji služe za proveru primenljivosti generatora slučajnih vrednosti u kriptografiji [8].

4. REALIZACIJA SIMULACIONOG MODELA

Da bi se zadržala jednostavnost softverskog rešenja i pritom omogućilo da implementacija bude: efikasna, lako nadogradiva i portabilna odabran je C++ programski jezik [9]. Svaka komponenta simulacionog modela je korisniku predstavljena interfejsom (abstraktnom klasom) sa osnovnim funkcionalnostima a implementirana u klasama koje ga nasleđuju. Na taj način je postignuta modularnost sistema i omogućena jednostavna nadogradnja novim funkcionalnostima.

4.1 LFRS

Osnova LFSR je 32bitna varijabla koja u svakom trenutku čuva vrednost registra sa metodama za: šiftovanje, računanje bita pariteta, čitanje izlazne vrednosti. LFSR je još definisan parametrima za: dužinu, inicijalnu vrednost, tip povratne sprege, bitskim maskama za povratnu spregu i izlaznu vrednost registra. Prelazak u naredno stanje se dobija računanjem bita pariteta, šiftovanjem za jedan korak u desno i setovanjem izračunatog bita na MSB polje.

Korisniku klase su na raspolaganju metode kojima se inicira prelazak u naredno stanje i metode za dobijanje vrednosti od interesa u svakom koraku.

4.2 Signali

Naponski i diter signali su predstavljeni klasama koje nasleđuju osnovni interfejs, različit za periodične, stohastičke i druge tipove funkcija. Obavezna je implementacija oblika vremenske funkcije kojom se dobija vrednost signala u zavisnosti od vremena.

4.3 Odabiranje

Odabiranje je izvedeno kompozicijom LFSR, naponskog i diter signala. Zasniva se na izračunavanju intervala odabiranja iz odnosa učestanosti. Izlaznim vrednostima iz LFSR se množi interval odabiranja i tako dobija vreme u kome odabiramo vrednost napona.

Različite modulacije f_s se dodaju novim nasleđivanjem osnovnog interfejsa i implementacijom stohastičke ili determinističke funkcije modulacije.

4.4 Konfiguracija i ispis

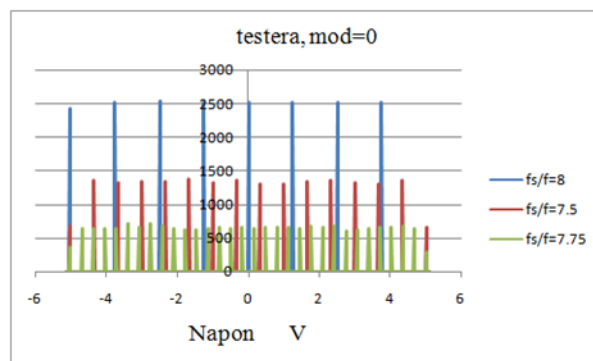
Implementacija simulacionog modela sadrži i rutine za ispis vrednosti u fajl, alate za bitsku manipulaciju i sistem za konfiguraciju same simulacije.

Pri pokretanju simulacije, učitava se konfiguracioni tekstualni fajl sa mapiranim imenima i vrednostima parametara (*key-value* struktura).

Konfiguracijom je omogućeno da se primenom *Abstract Factory* paterna [10] izabere jedna od postojećih implementacija za svaku komponentu, dužina registra, funkcija pariteta. Takođe je potrebno definisati oblik ispisa (binarni ili decimalni) i broj odbiraka koji će simulacija napraviti.

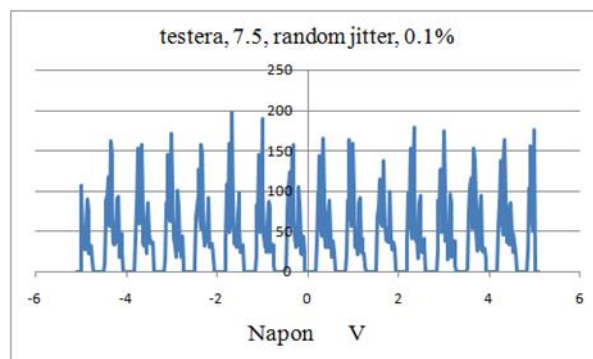
5. REZULTATI SIMULACIJA

Na slici 4. prikazan je histogram u slučaju testerastog talasnog oblika, bez modulacije učestanosti odabiranja, za tri vrednosti učestanosti odabiranja.



Slika 4. Histogram, bez modulacije učestanosti

Dobijamo diskretan skup vrednosti odbiraka. Broj mogućih vrednosti se ponaša u skladu sa teorijskim predviđanjem datim u Tabeli 1.

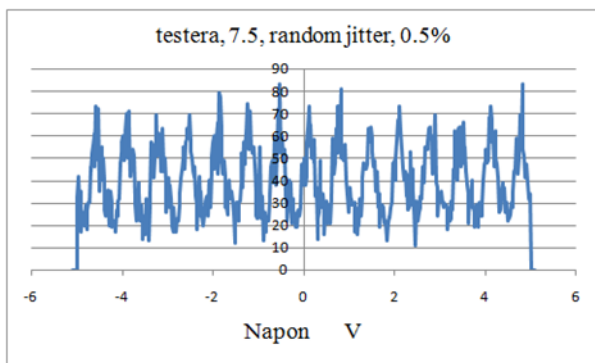


Slika 5. Histogram, stohastička modulacija, 0.1%

Na slici 5. je prikazan histogram vrednosti odbiraka pri srednjoj vrednosti odnosa učestanosti 7.5, sa stohastičkom modulacijom učestanosti odabiranja i indeksom modulacije 0.1 %. Uočava se opseg vrednosti umesto jedne diskretne vrednosti koja se dobija u slučaju bez modulacije. Opsezi nisu međusobno preklapljeni, i dalje postoje vrednosti koje se nikada ne dešavaju.

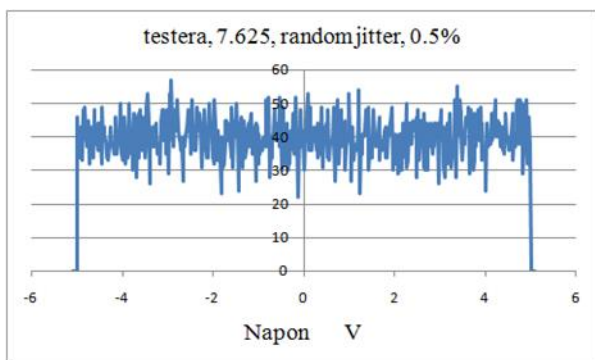
Na slici 6. je prikazan histogram vrednosti odbiraka pri srednjoj vrednosti odnosa učestanosti 7.5, sa stohastičkom modulacijom učestanosti odabiranja i indeksom modulacije 0.5 %.

Opsezi se sada preklapaju, sve vrednosti iz opsega testerastog napona su moguće, ali nisu sve vrednosti podjednako moguće.



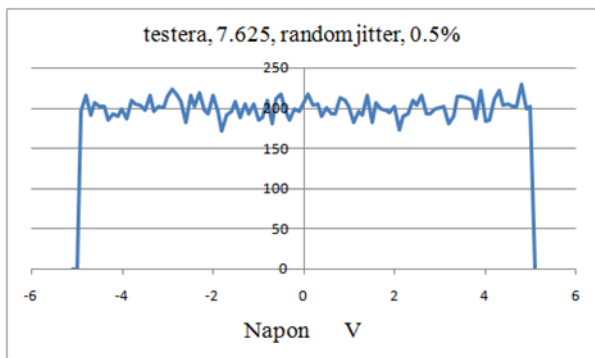
Slika 6. Histogram, stohastička modulacija, 0.5%

Na slici 7. je prikazan histogram vrednosti odbiraka pri srednjoj vrednosti odnosa učestanosti 7.625, sa stohastičkom modulacijom učestanosti odabiranja i indeksom modulacije 0.5 %.



Slika 7. Histogram za odnos učestanosti 7.625

Sada su sve vrednosti iz opsega testerastog napona moguće, ali ovoga puta sa približno jednakim verovatnoćama. Ista zavisnost se vidi na slici 8, ovaj put nacrtana sa deset puta manje tačaka po horizontalnoj osi.



Slika 8. Smanjen broj iscrtanih tačaka na x – osi

6. ZAKLJUČAK

Softverskom simulacijom modela su omogućeni neophodni koraci pri projektovanju hardverskog sistema: analiza, optimizacija i predviđanje ponašanja sistema pre stvarne implementacije. Omogućena je dobra polazna osnova za istraživanje predložene metode generisanja šuma.

Dodavanjem vizuelne prezentacije rezultata i podešavanja parametara bi se simulacija približila korisniku. Dodatnim matematičkim uopštavanjem konfiguracije komponenata modela bilo bi moguće dodavati nove implementacije bez softverske intervencije i programerskog predznanja.

7. LITERATURA

- [1] Vladimir Vujičić, Slobodan Milovančev, Mile Pešaljević, Ivan Župunski, "Low-Frequency Stochastic true RMS Instrument", IEEE Trans. Instr. Meas., vol 48, pp 467-470, April 1999.
- [2] Vladimir Vujičić, "Generalized Low-Frequency Stochastic True RMS Instrument", IEEE Transactions on Instrumentation and Measurement, Vol. 50., No. 5, October 2001.
- [3] D. Pejic, V. Vujicic, "Accuracy limit of high precision stochastic watt-hour meter", Instrumentation and Measurement, IEEE Transactions (Volume: 49, Issue: 3), June 2000, pp. 617-620, doi: 10.1109/19.850404.
- [4] Urekar Marjan, Gazivoda Nemanja V, Pejic Dragan, "The Core for High-Precision Stochastic Smart Grid Meter Based on Low-Resolution Flash ADC", IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, (2019), vol. 68 br. 6, str. 1705-1713
- [5] D. Pejić, N. Gazivoda, B. Ličina, M. Urekar, P. Sovilj, B. Vujičić, "A Proposal of a Novel Method for Generating Discrete Analog Uniform Noise", Advances in Electrical and Computer Engineering, Vol. 18, No. 3, doi: 10.4316/AECE.2018.03009, 2018.
- [6] J. Neškudla, J. Vedral, "Digitally controlled white noise generator", 2006. URL: <http://www.imeko.org/publications/wc-2006/PWC-2006-TC4-IWADC-008u.pdf> (31.01.2016.)
- [7] Donald E. Knuth, "The Art of Computer Programming", Volume 3, Sorting and Searching.
- [8] F. Pareschi, R. Rovatti, G. Setti, "On statistical tests for randomness included in the NIST SP800-22 test suite and based on the binomial distribution", IEEE Transactions on Information Forensics and Security, 7(2), pp.491-505, 2012.
- [9] Scott Meyers, "Effective C++", Addison-Wesley Professional; 3 edition (May 22, 2005)
- [10] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, "Elements of Reusable Object-Oriented Software", Addison-Wesley Professional; 1 edition (October 31, 1994)

Kratka biografija:



Marko Knežević rođen je u Užicu 1982. god. Radi kao programer u Novom Sadu. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva odbranio je 2019.god.

OBNOVLJIVI IZVORI ENERGIJE U ZAŠTIĆENIM PODRUČJIMA I POTENCIJAL KORIŠĆENJA SOLARNE ENERGIJE U MANASTIRU KOVILJ**RENEWABLE ENERGY SOURCES IN NATURE RESERVES AND THE POTENTIAL OF SOLAR ENERGY USE IN KOVILJ MONASTERY**

Dragana Stajić, Vladimir Katić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**2. ZAŠTIĆENA PODRUČJA SRBIJE**

Kratak sadržaj – U radu je predstavljen pregled obnovljivih izvora električne energije i njihova mogućnost primene u zaštićenim područjima Srbije. Dat je primer zaštićenog područja Koviljsko-Petrovaradinski rit i manastira Kovilj koji se nalazi u sklopu ovog područja, sa predlogom implementacije solarne energije na manastirskom imanju i tehno-ekonomskom analizom rešenja.

Ključne reči: Zaštićena područja, obnovljivi izvori energije, fotonaponski sistemi, solarna energija

Abstract – This paper gives an overview of renewable energy sources and the potential for their implementation in Serbia's nature reserves. It covers the example of Koviljsko-Petrovaradinski rit nature reserve and Kovilj monastery located in the reserve. The paper proposes a solution for the implementation of solar energy on the monastery estate along with the techno-economic analysis.

Keywords: Nature reserves, renewable energy sources photovoltaic systems, solar energy

1. UVOD

Zaštićena područja su predeone celine u kojima postoji geološka, biološka ili ekosistemska raznovrsnost i specifičnost, i koje su pod zaštitom države. Kako u zaštićenim područjima postoje mnogo rigorozniji kriterijumi kad je u pitanju zagađivanje životne sredine, od velikog je značaja da se omogući njihovo snabdevanje energijom iz obnovljivih izvora, ali tako da ovi izvori svojom konstrukcijom i radom ne narušavaju samu sredinu. U tom smislu najpogodnije je korišćenje solarne energije, ali isključivo ako se postave na postojeće objekte u obliku krovnih fotonaponskih (FN) elektrana za proizvodnju električne energije i/ili solarnih kolektora za dobijanje toplotne energije (topla voda).

U radu je posmatrano zaštićeno područje Koviljsko-Petrovaradinskog rita u blizini Novog Sada i manastir Kovilj koji se nalazi pored ovog zaštićenog područja. Razmatrana je upotreba solarne energije i dat predlog rešenja krovne FN elektrane, uz tehno-ekonomsku analizu rešenja i proračun uštede u emisiji CO₂.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Katić, red. prof.

U Srbiji postoji podela zaštićenih područja u tri kategorije: I kategorija (prirodna dobra od izuzetnog značaja), II kategorija (prirodna dobra od velikog značaja) i III kategorija (značajna prirodna dobra). Kriterijumi po kojima se zaštićena područja svrstavaju u jednu od ove tri kategorije su podeljeni u tri grupe: suštinska svojstva dobra, funkcija i značaj dobra, i ugroženost zaštićenog prirodnog dobra.

Postoji 7 vrsta zaštićenih područja u Srbiji: nacionalni park, park prirode, predeo izuzetnih odlika, opšti i specijalni rezervat prirode, spomenik prirode i zaštićeno stanište.

Po načelu prevencije i predostrožnosti, člana 9 Zakona o zaštiti životne sredine Republike Srbije, svaka aktivnost mora biti planirana i sprovedena na način da: prouzrokuje najmanju moguću promenu u životnoj sredini; predstavlja najmanji rizik po životnu sredinu i zdravlje ljudi; smanji opterećenje prostora i potrošnju sirovina i energije u izgradnji, proizvodnji, distribuciji i upotrebi; uključi mogućnost reciklaže; spreči ili ograniči uticaj na životnu sredinu na samom izvoru zagađivanja [1].

3. KOVILJSKO-PETROVARADINSKI RIT

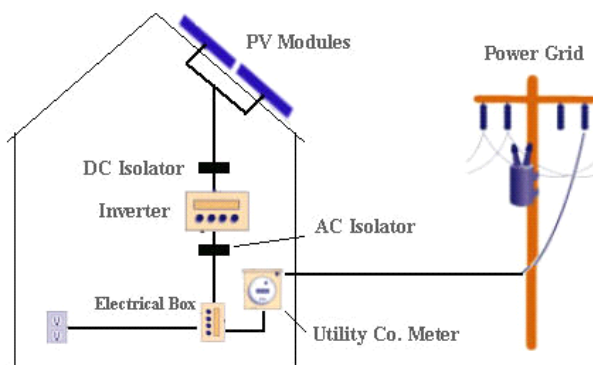
Specijalni rezervat prirode Koviljsko-Petrovaradinski rit se nalazi u jugoistočnoj Bačkoj i prostire na 5895 hektara površine. Ovo je naš najprezentativniji očuvani ritski kompleks. Unutar Koviljsko-Petrovaradinskog rita postoje područja sva tri stepena režima zaštite, od čega u režim zaštite I stepena spada 373 hektara odnosno 6,33% površine rezervata, u režim zaštite II stepena spada 1738 hektara odnosno 2,48% površine rezervata, i u režim zaštite III stepena spada 3784 hektara odnosno 64,19% površine rezervata.

Koviljsko-Petrovaradinski rit poseduje i nekoliko međunarodnih stepena zaštite, a to su: IBA (značajno područje za ptice od 1989. godine), IPA (značajno područje za biljke od 2005. godine), ICPDR (zaštićeno područje zavisno od vode i značajno za basen Dunava) i pripadnost spisku važnih područja Ramsarske konferencije od 2012. godine. Kako je na postojećim lokalitetima infrastruktura prilično bazična, za potrebe ovog rada uzet je Koviljski manastir, koji predstavlja izazov zbog svoje vrednosti kao kulturnog dobra čija se estetika ne sme narušiti prilikom implementacije novih energetskih izvora [2].

4. SOLARNA ENERGIJA

Prednost solarne energije je mala količina štetnih gasova koji se proizvode njenom upotrebom (srednja vrednost emitovanog CO₂ po kilovat-času iznosi 48 g za solarne sisteme kapaciteta 10 MW i većeg, odnosno 41 g za krovne solarne sisteme manjih kapaciteta). Solarni sistemi su tihi i ne uznemiravaju životinjski svet biosfere, a mogu se postavljati na postojeće objekte, čime se izgled zaštićenog područja minimalno menja. Nedostatak solarne energije je da proizvodnja zavisi od doba dana i vremenskih uslova. Takođe, inicijalna cena izgradnje sistema je visoka, i treba nekoliko godina dok se sistem ne isplati. Prema podacima Ministarstva energetike Srbije, prosek sunčevog zračenja u Srbiji je za oko 40% veći od evropskog proseka sa 1,4 kWh godišnje po m². [3].

Da bi se sunčeva energija uspešno konvertovala u električnu energiju, potrebno je da se koriste fotonaponski elementi. Ovi elementi predstavljaju poluprovodničke strukture koje pretvaraju sunčevu energiju u napon preko fotonaponskog efekta. Fotonaponski sistem je struktura koja povezuje generatore energije (solarne panele) sa potrošačem, uz potrebne elemente za konverziju. U fotonaponski sistem pored fotonaponskih ćelija spadaju i DC/AC konvertori (invertori) i propratna električna oprema koja služi za povezivanje sa električnom mrežom. Na slici 1 je prikazana šema mrežnog FN sistema sa bidirekcionim brojilom.



Slika 1. Šema mrežnog fotonaponskog sistema sa bidirekcionim brojilom [4]

5. PREDLOG ENERGETSKOG REŠENJA ZA MANASTIR KOVILJ

5.1 Razmatranje o mogućnostima primene OIE

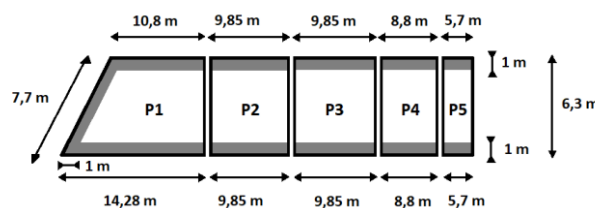
Objekti manastira ne spadaju u teritoriju rezervata već se nalaze neposredno pored, pa nema prepreka u smislu ograničenja vezanih za različite stepene zaštite. Međutim, fotonaponski paneli su velikih dimenzija i svojim izgledom bi mogli udroziti manastirski kompleks. Iz tog razloga je lokacija odabrana uz konsultacije sa manastirom.

Predlog rešenja je da se postave fotonaponski paneli na krovu južno-orijentisanog dela manastirskog kompleksa, gde je osunčanost najbolja, a položaj takav da bi paneli ne bi bili vidljivi posetiocima manastira i ne bi estetski narušavali objekat. Na slici 2 prikazana je odabrana

lokacija (slika 2.a), kao i dimenzije krovne površine dela koji je odabran za postavljanje FN panela (slika 2.b).



a)



b)

Slika 2. Lokacija FN elektrane i dimenzije krovnih površina

Izbor solarnog panela je pao na model BenQ SunVivo PM060MW2 dimenzija 1640x992x40 mm, maksimalne snage 300 W, dok je za noseću konstrukciju izabran sistem za krovove sa crepom proizvođača Solar First. Na predviđenu površinu krova moguće je postaviti 102 panela, tako da je ukupna instalisana snaga 30,6 kWp.

Kod izbora invertora, uzeto je u obzir da njegova snaga mora biti u rasponu $\pm 10\%$ projektovane snage sistema, to jest u rasponu od 27,5 kW do 33,6 kW. Za idejno rešenje je izabran trofazni inverter TRIO-27.6-TL-OUTD-S2F-400 proizvođača ABB snage 27,6 kW.

Za nadzemni kabl je izabran model Solar 125 S flex UL/EN proizvođača BETAflam, dok je za podzemni kabl izabran model PP 00 4x6 proizvođača Novkabel.

Solarna FN elektrana predviđena je da bude direktno priključena na mrežu (*In-grid*), odnosno da se sva proizvedena energija isporučuje EES-u. Manastir bi mogao da prihoduje, ako bi se za isporučenu energiju koristila povlašćena (*feed-in*) tarifa ili na bazi kompenzacije.

5.2 PVGIS proračun

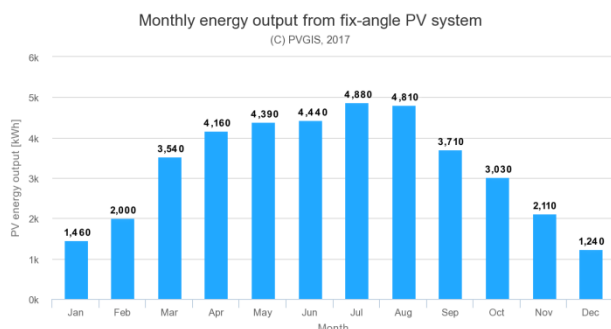
Za proračun je korišćen softver Photovoltaic Geographical Information System (PVGIS) koji uzima u obzir intenzitet sunčevog zračenja po mesecima u zavisnosti od geografskih koordinata, i omogućava procenu proizvodnje sistema u toku jedne godine. Unošenjem geografskih koordinata manastira i ostalih podataka (snaga, položaj, nagnutost i sl.), dobija se da je godišnja proizvodnja energije PV sistema 39.800 kWh, dok je vrednost godišnjeg sunčevog zračenja 1580 kWh/m². Procenjeni gubici sistema na osnovu softverske simulacije: gubici zbog upadnog ugla svetlosti: -2,8%, spektralno rasejanje: 1,1%, gubici zbog temperature i niskog zračenja: 6,8%, ukupni gubici: -17,6%. U tabeli 1 su prikazane vrednosti prosečna mesečna proizvodnja električne energije sistema (Em) u kWh/m², prosečnog

sunčevog zračenja na FN modul (Hm) u kWh/m² i standardne devijacije (SDm) u kWh.

Tabela 1. Mesečna proizvodnja električne energije, sunčevo zračenje i standardna devijacija

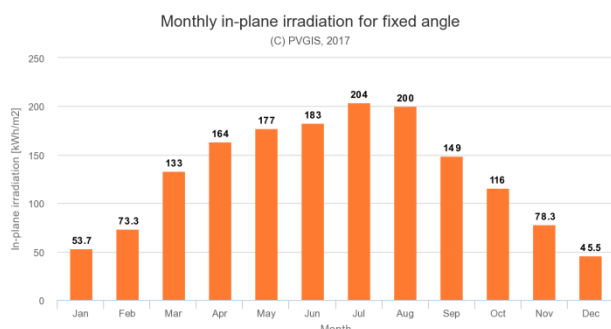
Mesec	Em	Hm	SDm
Januar	1460	53,7	282
Februar	2000	73,4	306
Mart	3540	134	487
April	4160	164	504
Maj	4390	177	297
Jun	4440	183	332
Jul	4880	204	293
Avugust	4810	200	250
Septembar	3710	149	500
Oktobar	3030	117	383
Novembar	2110	78,3	219
Decembar	1240	45,6	294

Proizvodnja električne energije dostiže svoj maksimum od 4.880 kWh u toku meseca jula, dok je očekivani minimum proizvodnje u toku decembra, i iznosi 1.240 kWh, što je prikazano na slici 3.



Slika 3. Mesečna proizvodnja el. energije PV sistema

Nivo sunčevog zračenja dostiže maksimum od 204 kWh/m² u julu, dok je najniži tokom decembra i iznosi 45,5 kWh/m², što je prikazano na slici 4.



Slika 4. Mesečno sunčevo zračenje na PV sistem

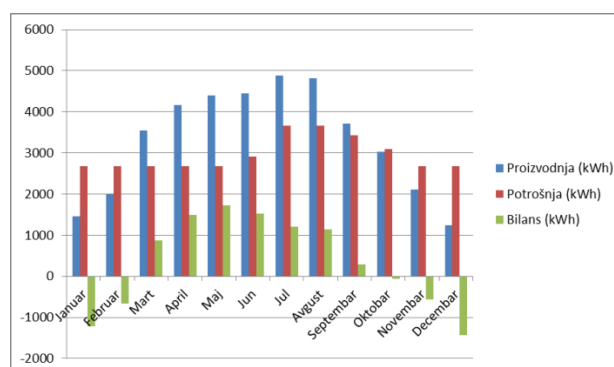
5.3 Tehno-ekonomska analiza

Uzevši u obzir okvirne troškove opreme i održavanja projektovane solarne elektrane, potrebno je uložiti 24.561 €. U tabeli 2 su prikazane cene i količina potrebne opreme kao i ostali troškovi, koji odlaze na godišnje održavanje sistema.

Tabela 2. Lista opreme potrebne za izgradnju elektrane

vrsta opreme	cena po jedinici (€)	količina	ukupno (€)
solarni panel	182	102	18564
inverter	3499	1	3499
noseća konstrukcija	1500	1	1500
nadzemni kabl	1,09	450	490.5
podzemni kabl	3,15	50	157.5
razvodni orman	200	1	200
ostali troškovi	/	/	150
UKUPNO			24561

Kako konkretna godišnja potrošnja električne energije u manastiru nije poznata, napravljena je procena na osnovu broja i vrste potrošača. Dobijeno je da manastir godišnje potroši 35.479,2 kWh, što znači da se uz proračunatu proizvodnju električne energije u solarnoj elektrani godišnje proizvede višak od 4.290,8 kWh. Na slici 5 je prikazan odnos proizvedene i utrošene električne energije po mesecima.

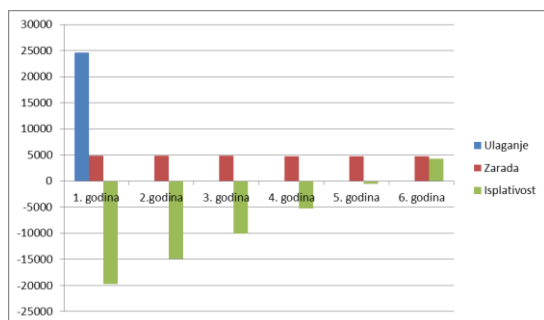


Slika 5. Grafički prikaz bilansa mesečne proizvodnje i potrošnje električne energije

Nakon analize dva slučaja: prodaje viška proizvedene električne energije elektrodistribuciji po podsticajnoj ceni, i prodaje celokupne proizvedene električne energije elektrodistribuciji po podsticajnoj ceni, utvrđeno je da je prva opcija isplativija. Prilikom proračuna isplativosti je uzeto u obzir da solarni paneli godišnje gube 0,5% snage, te da će srazmerno tome godišnja proizvodnja električne energije biti manja iz godine u godinu. Takođe su uzeti u obzir godišnji troškovi održavanja elektrane koji iznose 150 evra godišnje, i ušteda na računima za električnu energiju koji su procenjeni na 4487,1 € godišnje zbog velikih potrošača u vidu opreme za proizvodnju sveća, sira, meda i voćnih prerađevina. U tabeli 3 je prikazana isplativost elektrane u slučaju prodaje viška električne energije. Na slici 6 je prikazano da se sistem isplati nakon 6 godina rada, uz podsticajnu cenu od 0,121 c€/kWh.

Tabela 3. Isplativost solarne elektrane uz prodaju viška proizvedene električne energije

Godina	Godišnja zarada na proizvedenoj el. energiji (€)	Godišnji troškovi održavanja (€)	Godišnja ušteda na računima (€)	Ukupna godišnja zarada (€)	Isplativost (€)
1. godina	523,3	150	4487,1	4860,4	-19700,6
2. godina	499	150	4487,1	4836,1	-14864,4
3. godina	474,8	150	4487,1	4811,9	-10052,5
4. godina	450,5	150	4487,1	4787,6	-5264,9
5. godina	426,3	150	4487,1	4763,4	-501,5
6. godina	402	150	4487,1	4739,1	4237,6

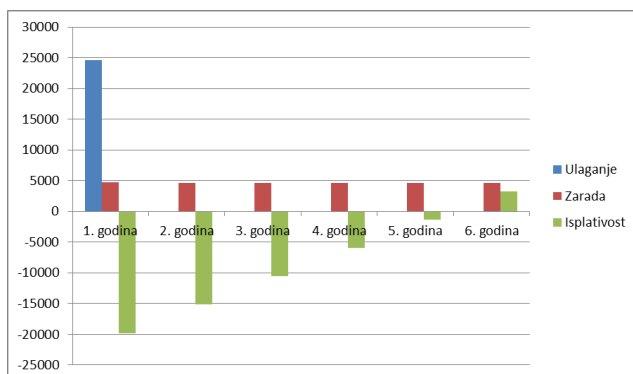


Slika 6. Grafički prikaz isplativosti elektrane pri prodaji viška el. energije po podsticajnoj ceni

U drugom posmatranom slučaju, solarna elektrana se u potpunosti koristi za prodaju električne energije, te je zarada na prodaji mnogo veća, ali nema uštede na računima koji iznose jednako kao i pre implementacije solarne elektrane. Na slici 7 je prikazana isplativost elektrane pri prodaji celokupne proizvedene energije.

Tabela 4. Isplativost solarne elektrane uz prodaju celokupne proizvedene električne energije

Godina	Godišnja zarada na proizvedenoj el. energiji (€)	Godišnji troškovi održavanja (€)	Godišnja ušteda na računima (€)	Ukupna godišnja zarada (€)	Isplativost (€)
1. godina	4850,3	150	0	4700,3	-19860,6
2. godina	4826,1	150	0	4676,1	-15184,5
3. godina	4801,8	150	0	4651,8	-10532,7
4. godina	4777,6	150	0	4627,6	-5905,1
5. godina	4753,3	150	0	4603,3	-1301,8
6. godina	4729,1	150	0	4579,1	3277,3



Slika 7. Prikaz isplativosti elektrane pri prodaji celokupne proizvedene el. energije

Iako bi, zbog podsticajne cene, bilo logično pretpostaviti da je ova varijanta isplativija, to nije slučaj zbog visokih mesečnih računa za električnu energiju. Kako manastir prelazi granicu od 1600 kWh mesečne potrošnje, ulazi u crvenu tarifnu zonu gde je cena 0,13 c€/kWh, što je za 0,01 c€ više od podsticajne cene. Rešenje bi bio prelazak na višetarifno merenje, čime bi, korišćenjem pogodnosti niže tarife, računi bili znatno manji, pa bi opcija prodaje celokupne proizvedene električne energije bila isplativija.

5.4 Ušteda CO₂

Na osnovu Izveštaja o stanju životne sredine u JP "Elektroprivreda Srbije" za 2018. godinu[5], u TE "Nikola Tesla" je u toku 2018. godine proizvedeno ukupno 16.614,7 GWh električne energije, a emitovano je

19.341.895 tona CO₂, što znači da ova TE godišnje emituje 1,164 kg/kWh CO₂. Kako projektovani solarni sistem godišnje proizvodi 39.800 kWh, uštedelo bi se ukupno 46327,2 kg CO₂ na godišnjem nivou prelaskom na solarni sistem napajanja.

6. ZAKLJUČAK

Primena obnovljivih izvora energije u zaštićenim sredinama predstavlja izazov zbog raznih ograničavajućih faktora koji se moraju uzeti u obzir prilikom analize. U radu je razmatrana mogućnost instalacije solarne elektrane na način da estetski ne narušava izgled manastira Kovilj, i utvrđeno da je moguće napraviti elektranu snage 39.800 kWh koja bi dala pozitivan godišnji bilans od 4290,8 kWh.

Zbog male osunčanosti u zimskim mesecima, proizvodnja elektrane nije dovoljna da pokrije potrošnju, te je neophodno osloniti se na struju iz elektrodistribucije. Razmatrana su dva slučaja: prodaja celokupne proizvedene električne energije, i prodaja samo viška proizvedene energije. U oba slučaja, elektrana se samoispplaćuje nakon šest godina upotrebe, uz podsticajne cene fid-in tarife. Iako je u ovom radu posmatrana samo upotreba solarne energije, postoji prostor za dalje analiziranje primene obnovljivih izvora energije kroz upotrebu geotermalnih pumpi.

7. LITERATURA

- [1] Zakon o zaštiti životne sredine, Sl. glasnik RS, br. 135/2004, 36/2009
- [2] Specijalni rezervat prirode "Koviljsko-Petrovaradinski rit", predlog za stavljanje pod zaštitu kao zaštićenog područja I kategorije, studija zaštite, Pokrajinski zavod za zaštitu prirode, 2010.
- [3] "Fotonaponski sistemi", Studija NPPE, evidencioni broj EE704-1052A, Beograd 2004.
- [4] "Grid-Connected Power Systems", Murdoch University, 2008.
- [5] Izveštaj o stanju životne sredine u JP „Elektroprivreda Srbije“ za 2018. godinu.

Kratka biografija:



Dragana Stajić rođena je u Novom Sadu 1984. god. Diplomsko-master studije na studijskom programu Energetika, elektronika i telekomunikacije upisala je školske 2003/2004. Na studijama se opredelila za modul Mikroracunarska elektronika. Master rad je odbranila 2019. godine.



Vladimir A. Katić, red.prof. rođen je 1954. godine u Novom Sadu. Doktorirao je na Univerzitetu u Beogradu 1991. godine. Za redovnog profesora Univerziteta u Novom Sadu izabran je 2002. god. Oblasti interesovanja su mu energetska elektronika, obnovljivi izvori električne energije, kvalitet električne energije i električna vozila.

APLIKACIJA ZA PRAĆENJE CENA PO MARKETIMA**APPLICATION FOR PRICE TRACKING BY MARKET**

Aleksandar Ljahović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Ovaj članak treba da demonstrira probleme koji se javljaju tokom implementacije aplikacije za praćenje cena po marketima i način na koji su oni rešeni. Takođe, pojašnjava se i tehnologije u kojima je implementirana aplikacija i pojmovi elektronske trgovine i elektronskog poslovanja.

Ključne reči: e-trgovina, ASP.NET Web API, Angular

Abstract – This article should demonstrate the problems that appear during implementation of the application for price tracking by markets and the way they are solved. Also, the technologies used for application implementation and terms of e-commerce and e-business will be explained.

Keywords: e-commerce, ASP.NET Web API, Angular

1. UVOD

U članku se opisuje napravljena aplikacija koja prikazuje najnižu cenu proizvoda u svim prodavnicama koje prodaju taj proizvod, značaju takvih aplikacija i način na koji je ona implementirana.

U prvoj celini opisan je pojam elektronskog poslovanja, zašto je ovakvo poslovanje bitno i šta doprinosi preduzećima. Takođe, opisan je i definisan pojam elektronske trgovine. Na kraju ove celine opisana je aplikacija koja je razvijana, funkcionalnosti koje podržava i ograničenja koja su uvedena.

Kroz drugu celinu se opisuju tehnologije, radni okviri i prakse koje su korišćene pri implementaciji aplikacije.

U trećoj celini opisan je način na koji je aplikacija implementirana. U ovoj celini opisana je implementacija onoga što korisnik vidi, informacije koje su mu od interesa, informacije koje sme da vidi u zavisnosti od uloge koju ima, mogućnost unosa, izmene i brisanja podataka i slično.

2. OPIS REŠAVANOG PROBLEMA

Pod pojmom elektronsko poslovanje (e-poslovanje, eng. **E-business**) podrazumeva se kupovina i prodaja na internetu, organizovanje poslovne komunikacije prema klijentu, organizacija poslovanja firme u mrežnom okruženju, briga o klijentu,...

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Aleksandar Kupusinać, van. prof.

Pod elektronskim poslovanjem podrazumeva se obavljanje poslovnih procesa uz primenu elektronske tehnologije. Elektronska tehnologija podrazumeva kombinovanu upotrebu informacionih tehnologija i telekomunikacija. Ova vrsta tehnologije omogućava slanje velikog broja informacija, na velike daljine u kratkom vremenskom periodu. To omogućava preduzeću, koje u svom poslovanju koristi elektronsku tehnologiju, da ostvari značajne uštede u troškovima poslovanja, efikasnije obavlja svoje zadatke i, samim tim, bude konkurentnije na tržištu. E-poslovanje omogućava stvaranje potpuno novog modela poslovanja, prilagođavanje kupcu, neprekidan pristup svetskom tržištu, efikasnije poslovanje, smanjivanje troškova itd.

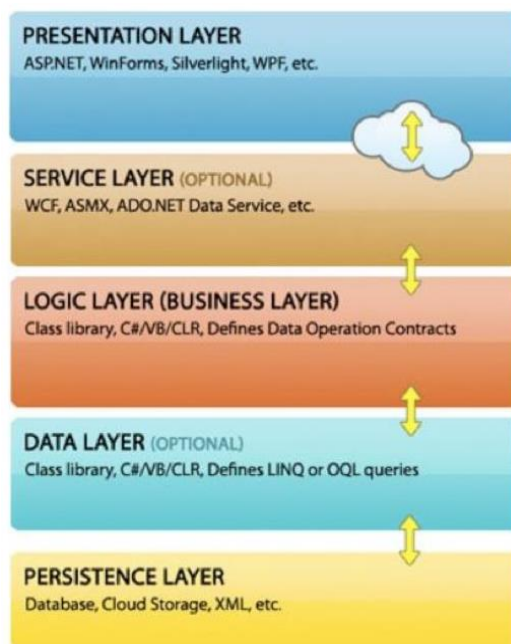
Elektronska trgovina (E-trgovina) je multidisciplinarni koncept, koji utiče na načine na koje se odvijaju interakcije i pregovori sa klijentima, načine na koje se obavljaju plaćanja, kao i odnose sa dobavljačima. E-trgovina se može posmatrati sa šireg i užeg stanovišta pa tako šira definicija obuhvata razmenu poslovnih informacija, održavanje poslovnih odnosa i vođenje poslovnih transakcija sredstvima telekomunikacionih mreža, a uža definicija obuhvata kupovinu i prodaju robe, usluga i informacija putem mreže. Prednosti elektronske trgovine u odnosu na tradicionalnu su mnogobrojne i značajne su kako za kupce i prodavce, tako i za čitavo društvo. Elektronska trgovina smatra se globalnim fenomenom i kao takva podržava nekoliko modela. Dobra stvar kod e-trgovine je mogućnosti korišćenja jednog ili kombinovanje više modela kako bi poslovanje bilo uspešnije. Poslovni modeli, ili vrste, e-trgovine mogu se generalno svrstati nekoliko kategorija: biznis prema biznisu (B2B), biznis prema potrošaču (B2C), potrošač prema potrošaču (C2C), biznis prema vladi (B2G), vlada prema biznisu (G2B), vlada prema potrošaču (G2C).

Aplikacija za praćenje cena po marketima u osnovi treba da omogući korisnicima pregled cena po raznim marketima za neki proizvod i da mu izdvoji najpovoljniju cenu za taj proizvod. Aplikaciju mogu koristiti 3 tipa korisnika (administratori, kupci i posmatrači). Zadatak administratora je održavanje sajta. Ovaj korisnik radi održavanja sajta ima mogućnost dodavanja novih i brisanja postojećih proizvoda, prodavnica i kategorija proizvoda. Kupac je korisnik koji pored mogućnosti da vidi cene proizvoda po marketima, kao i najbolju cenu na početnoj stranici može da dodaje i uklanja proizvode iz korpe i prati najnižu cenu za odabrane proizvode. Proizvode iz korpe kupac može kupiti ili ih izbaciti iz nje. Posmatrač je korisnik aplikacije koji nema mogućnost rada sa korpom (dodavati, brisati, pregledati niti kupovati

proizvode), ali može pratiti cene po marketima. Administratori i kupci se moraju logovati kako bi izvršavali dodatne aktivnosti. Takođe, oni imaju mogućnost izmene svojih profila. Početna stranica sadrži nekoliko izlistanih proizvoda i mogućnost promene strane kako bi se prikazali neki drugi proizvodi. Na početnoj stranici je moguće i pretraživanje proizvoda po nazivu, kao i filtriranje po kategoriji kojoj pripadaju.

3. OPIS KORIŠĆENIH TEHNOLOGIJA I ALATA

Aplikacija koja prati cene po marketima organizovana je po višeslojnoj arhitekturi. Ovaj stil arhitekture se najčešće svede na troslojnu arhitekturu koja obuhvata prezentacioni sloj, sloj poslovne logike i sloj podataka.



Slika 1. Višeslojna arhitektura aplikacija.

Višeslojna arhitektura ima mnogo prednosti zbog čega se, iako nastala početkom 90-ih, i dalje najčešće koristi u razvijanju aplikacija. Prezentacioni sloj je zadužen za komunikaciju sa krajnjim korisnikom. U njemu se izrađuje deo aplikacije koji korisnik treba da vidi. Servisni sloj služi za razmenu podataka između prezentacionog i sloja poslovne logike.

Ovaj sloj je opcioni pošto se često spaja sa slojem poslovne logike. Neki od zadataka ovog sloja su obrada zahteva, obaveštavanje o greškama i vođenje računa o sigurnosti sistema. Sloj poslovne logike implementira poslovne procese i poslovne komponente aplikacije. Zadužen je za obradu i validiranje svih zahteva sistema, sadrži logiku i poslovna pravila koja održavaju aplikaciju u celini. Sloj za pristup podacima ima zadatak da ostatku sistema prikupi zahtevane podatke, da ih izmeni ili upiše u skladište podataka koje aplikacija koristi. Samo ovaj sloj treba da ima pristup skladištu i da zna njegovu strukturu podataka. Sloj za perzistenciju predstavlja skladište u koje se smeštaju podaci koji su potrebni sistemu za funkcionisanje. U ova skladišta podataka moguće je upisati, izbaciti, izmeniti i iščitati podatke u zavisnosti šta je zahtevano od sloja iznad. Ovaj sloj podrazumeva bazu podataka, cloud, XML i slično.

U implementaciji prezentacionog sloja korišćen je Angular radni okvir. Angular je JavaScript radni okvir (eng. framework) otvorenog koda koji se koristi za razvoj klijentskih web aplikacija. Za izradu aplikacija u Angular-u najčešće se koristi TypeScript. To je jezik otvorenog koda koji predstavlja nadskup JavaScript-a. JavaScript program je i TypeScript program, dok obrnuto ne mora da važi. TypeScript se kompajlira u JavaScript i dizajniran je za razvoj velikih aplikacija. Angular projekat moguće je kreirati ručno, koristeći šablon projekta ili Angular CLI. Za razliku od prve dve opcije, Angular CLI omogućava ne samo brzo kreiranje projekta nego i komande za njegov brzi razvoj. Umesto da programer razmišlja o uvezivanju novih elemenata u projektu, ove komande to automatski rade.

Veći deo razvoja u Angular aplikacijama vrši se u komponentama. Komponente kontrolišu prikaz jednog dela ekrana. U osnovi, one su zapravo klase koje preko metoda i svojstava (eng. **properties**) API-ja komuniciraju sa prikazom. Za grupisanje Angular komponenti, servisa, direktiva i **pipes**-ova koji se odnose na aplikaciju koriste se moduli. Moduli mogu uvoziti funkcionalnosti koje su izveze od drugih modula, kao i izvoziti svoje funkcionalnosti koje mogu koristiti drugi moduli. Servisi u Angular-u se koriste za pristup metodama i svojstvima drugih komponenti u celom projektu. Pomoću njih je moguće da istim podacima pristupa i manipuliše više komponenti u projektu. Servisi se mogu koristiti i da bi se komponente rasteretile pa se na primer validacije korisničkog unosa rade u njima, a komponentama signalizira ukoliko unos nije validan. Za dinamičko menjanje stranice i sadržaja na njoj, u Angular-u, se koristi rutiranje. Kod rutiranja povezuju se putanje (URL-ovi) sa odgovarajućim komponentama koje se učitavaju kada je njihova putanja pozvana. Vezivanje podataka (eng. **Data binding**) omogućava komunikaciju između komponenti i **template**-a tj. HTML-a. Njegova uloga je da neki sadržaj na HTML stranici učini dinamičkim.

U radu sa bazom aplikacija se oslanja na MySQL koji je višenitni i višekorisnički SQL sistem za upravljanje bazama podataka. Sistem radi kao server, obezbeđujući višekorisnički interfejs za pristup bazi podataka. Biblioteke za pristup bazi podataka MySQL postoje za većinu programskih jezika, čiji oblik zavisi od datog programskog jezika. ER (**entity-relationship**) dijagrami pokazuju odnose između entiteta. Najčešće se koriste za organizovanje podataka unutar baze podataka ili informacionih sistema.

Za razvoj sloja poslovne logike aplikacije korišćen je .NET Core. .NET Core na različitim platformama i platformama sa otvorenim kodom obezbeđuje lagani programerski model i fleksibilnost za rad na velikom broju razvojnih alata OS platformi. .NET Core se odnosi na više tehnologija koje obuhvataju .NET Core, ASP.NET Core i Entity Framework Core. On je kreiran da bi se .NET mogao koristiti na više platformi i u okruženjima koja imaju veća ograničenja. ASP.NET Core je **web** radni okvir kreiran od strane Microsoft-a za izradu **web** aplikacija, mikroservisa i API-ja. ASP.NET Web API je radni okvir koji olakšava izgradnju HTTP servisa koji dopiru do širokog kruga klijenata, uključujući

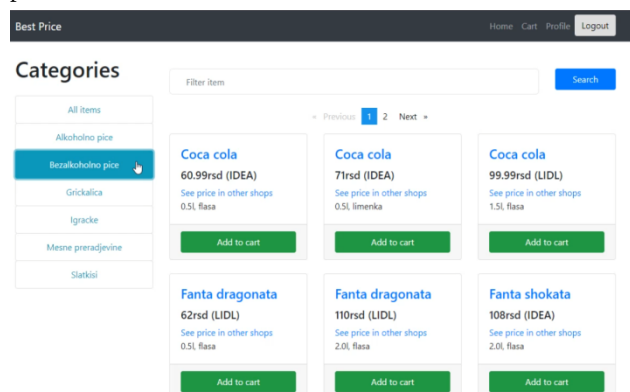
pretraživače i mobilne uređaje. Web API se posmatra kao HTTP interfejs koji od klijentske aplikacije prihvata HTTP zahtev i obavlja neke operacije nad podacima. Entity Framework (EF) Core je proširena verzija popularnog Entity Framework-a, tehnologije za pristup podacima. EF Core je verzija sa otvorenim kodom koja je podržana na više platformi. EF Core može da posluži kao **object-relation mapper** (ORM), omogućavajući .NET programerima da rade sa bazom podataka koristeći .NET objekte i eliminišu potrebu za većinom koda za pristup podacima koji obično treba da napišu.

Bootstrap je radni okvir namenjen izradi **front-end** dela **web** aplikacija. On je besplatan radni okvir otvorenog koda koji je baziran na HTML-u, CSS-u i JavaScript-u. Ovaj radni okvir olakšava izgradnju **web** stranica jer koristi globalne CSS postavke i osnovne HTML elemente, stilizovane i poboljšane proširenim klasama. Stranice napravljene u Bootstrap-u dostupne su na različitim platformama, za sve veličine ekrana. Dizajn **web** stranica je potpuno dinamičan i prilagodljiv (RWD - Responsive Web Design).

4. OPIS REŠENJA PROBLEMA

Implementacija aplikacije za praćenje cena u marketima i pronalaženje najniže cene bazirana je na troslojnoj arhitekturi. Za skladištenje podataka korišćena je MySQL baza podataka, a za manipulisanje bazom alat MySQL Workbench. Ovim alatom napravljena je šema sa tabelama, kolonama i relacijama između tabela. Relacijama između tabela postignuta su neka ograničenja: Svaka prodavnica može imati više proizvoda i svaki proizvod pripada tačno jednoj prodavnici. Dakle, isti proizvod u različitim prodavnicama posmatra se kao različit entitet. Ukoliko dođe do uklanjanja prodavnice iz baze, uklanjaju se i svi proizvodi koje je ona sadržala što je obezbeđeno opcijom kaskadnog brisanja.

Proizvod se svrstava u jednu kategoriju, dok jednoj kategoriji može pripadati više proizvoda. Korisnik može da bude ili administrator aplikacije ili kupac (registrovani korisnik). Administratorski nalog nema pravo kupovine pa mu ne treba ni korpa. Dakle, korisnik može a ne mora da ima svoju korpu u zavisnosti od uloge koju ima u aplikaciji. Korpa može biti prazna ali može sadržati i više proizvoda.



Slika 2. Početna stranica aplikacije za praćenje cena po marketima

Sprega između sloja poslovne logike i sloja za pristup podacima je generičko skladište čiji je zadatak da poveže

zahteve kontrolera sa bazom. Dakle, ovo skladište kada dobije zahtev od kontrolera treba da ga prilagodi i prosledi ka bazi tj. sloju za pristup podacima. Ovaj deo se najčešće implementira upotrebom **repository** šablona i može se posmatrati i kao deo sloja za pristup podacima. Interfejs IRepositoryBase koristi se kao osnova za implementaciju repository šablona i sadrži metode koje će koristiti sva konkretna skladišta. Konkretna skladišta se odnose na neki model ili tabelu u bazi podataka.

U konkretnim skladištima implementiraju se samo metode koje su od interesa za kontrolere. Kako se u ovim skladištima implementiraju kompleksnije metode kontroleri ostavljaju čitkiji i lakši za održavanje. Za razliku od čitanja iz baze, kod zahteva za manipulisanje bazom neophodno je uraditi i čuvanje tih izmena pa se kod kreiranja, izmena i brisanja mora eksplicitno pozvati čuvanje izmena u bazi. Postoje kontroleri kojima su potrebne informacija ili treba da manipulišu sa više entiteta iz baze. Da se u takvim kontrolerima ne bi morala instancirati sva, potrebna, konkretna skladišta, napravljen je omotač koji omogućava pristup svim skladištima.

Ovaj omotač kontroleri dobijaju **dependency injection**-om i koriste njegov interfejs za pristup konkretnom skladištu. Kao što je ranije rečeno, kontroleri u sloju poslovne logike imaju zadatak da obrade zahteve klijenta pristigle preko prezentacionog sloja, a zatim da preko istog vrati odgovor. Svaka metoda koja se isporučuje klijentu ima rutu koju je neophodno pogoditi da bi joj se pristupilo. Sve metode jednog Web API kontrolera imaju zajednički deo rute i deo rute koji se odnosi samo na tu metodu. Metode ovih kontrolera imaju i dekorator HTTP akcije koja se izvršava. Kada kontroler obradi pristigli zahtev, kroz interfejs IActionResult vraća odgovor klijentu. Ovaj interfejs očekuje odgovor u formatu status odgovora uz odgovarajuću poruku (objekat ili tekstualni opis). Kontroleri su, takođe, odgovorni i za validaciju modela. Kada sa prezentacionog sloja stigne zahtev za kreiranje ili izmenu nekog entiteta, kontroler će pre obrade tog zahteva proveriti validnost pristiglog objekta sa odgovarajućim modelom. Neretko se dešava da model kreiran u sloju za pristup podacima ne sadrži sve informacije koje kontroler treba da vrati prezentacionom sloju. U ovakvim situacijama pravi se DTO model.

U implementaciji prezentacionog sloja fokus je na prikazu svih funkcionalnosti korisnicima koji će koristiti aplikaciju, ali i pribavljanju neophodnih podataka sa servera. Prezentacioni sloj implementiran je korišćenjem Angular framework-a pa su za pribavljanje podataka zaduženi Angular servisi. Servisi imaju zadatak da preko kontrolera sloja poslovne logike komuniciraju sa serverskim delom aplikacije tako što će slati zahteve koje je korisnik inicirao ili zahteve za pribavljanje informacija koje su prezentacionom sloju od interesa. Kao što je opisano u kratkoj specifikaciji projekta aplikacija je namenjena za korišćenje različitih tipova korisnika. Različiti korisnici imaju različite uloge što znači i mogućnosti u radu sa aplikacijom. Veći deo početne strane aplikacije se ne menja u zavisnosti od korisnika. Svi korisnici imaju mogućnost da pretražuju proizvode po nazivu, kao i da ih filtriraju po kategoriji kojoj pripadaju. Na početnoj stranici prikazano je 6 proizvoda i mogućnost

listanja strana korišćenjem broja strane na koju korisnik želi da ode ili Previous/Next opcija za izlistavanje prethodne/sledeće strane proizvoda. Jedna od retkih razlika na početnoj strani je opcija dodavanja proizvoda u korpu. Korisnici sa ulogom kupca imaju mogućnost dodavanja, dok je ostalim dugme onemogućeno. Na klik „See price in other shops“ prikazuje se modalni dijalog u kojem su izlistane cene tog proizvoda u svim prodavnicama, sortirane po najpovoljnijoj. Modalni dijalog sadrži **header** (prikazuje informaciju o nazivu proizvoda), telo u kojem su izlistane prodavnice i cene za taj proizvod i **footer** na kojem se nalazi dugme za napuštanje dijaloga.

Otvaranjem stranice Manage administratoru su izlistane sve kategorije koje postoje u sistemu (bazi padataka). Administrator može da obriše kategoriju samo pod uslovom da je prazna (ni jedan proizvod joj ne pripada). Takođe, on može i da dodaje nove kategorije, a dovoljno je samo da unese njihov naziv. Osim kategorija, admin može da vidi sve prodavnice i sve proizvode. Za uklanjanje prodavnice iz sistema ne postoji ograničenje za proizvode kao kod kategorija (sme biti vezana za proizvode) već se zajedno sa njom uklanjaju i svi njeni proizvodi. Brisanje pojedinačnih proizvoda iz prodavnice je takođe moguće bez ikakvih ograničenja. Admin pri dodavanju prodavnice pored naziva unosi i adresu na kojoj se ona nalazi. Pri dodavanju proizvoda admin, između ostalog, mora odabrati kategoriju i prodavnicu kojima proizvod pripada.

Stranica Cart tabelarno prikazuje sve proizvode koje je kupac dodao u korpu. U tabeli su osim naziva proizvoda prikazane i najniže cene i prodavnica u kojoj je ta cena aktuelna. Takođe, pored svakog proizvoda nalazi se dugme „Cancel“ koje omogućava korisniku da iz korpe izbaci taj proizvod. Pored pojedinačnog izbacivanja proizvoda, kupac na „Cancel All“ dugme može da izbaci sve proizvode tj. da isprazni korpu.

Kupcu je takođe prikazana informacija o ukupnoj, najnižoj, vrednosti svih odabranih proizvoda. Za kraj, kupac ima mogućnost kupovine proizvoda pritiskom na dugme „Buy“. Admin i kupac otvaranjem stranice Profile mogu videti neke od podataka koje su uneli kao što su korisničko ime, ime i prezime, **email** adresa, broj mobilnog i odabrana slika. Oni mogu da promene svoju sliku odaberom druge slike sa svog računara. Osim slike, korisnici mogu menjati i druge podatke (ime, prezime, broj mobilnog). Takođe, ukoliko su napravili izmene, a žele da ih ponište, postoji dugme „Reset“ kojim se poništavaju sve unete izmene. Admini i kupci mogu na stranici Profile menjati i svoju lozinku tako što će u polja „Password“ i „Verify“ uneti novu lozinku (lozinka se mora poklapati u oba polja kako bi bila prihvaćena). Pored nabrojanih stranica, postoji i Login stranica preko koje se admini i kupci autentifikuju na sistemu. Kako bi se ulogovali, neophodno je da unesu korisničko ime ili **email** u prvo polje i odgovarajuću lozinku u drugo polje.

5. ZAKLJUČAK

U implementaciji aplikacije korišćene su tehnologije koje su jedne od najpopularnijih u svetu. Angular polako gubi bitku sa React-om koji je dosta jednostavniji pa se sve

više koristi, međutim Angular je sve bolji iz verzije u verziju i ima mnoštvo prednosti u odnosu na ostale radne okvire za izradu front-end dela aplikacije. MySQL vodi borbu sa Oracle i Microsoft SQL Server, a ASP.NET je i uz porast popularnosti i dalje iza PHP.

Pri implementiranju su korišćene dobre prakse kao što su primena višeslojne arhitekture i podela nadležnosti, primenjivani su dizajn šabloni, pravljeni omotači (eng. **wrappers**) itd. Ovakve aplikacije mogu mnogo pomoći kupcima da povoljnije izvrše kupovinu. Umesto da pamte cene po prodavnicama ili ulaze na razne sajtove prodavnica da bi uporedili cene, ova aplikacija ima sve cene na jednom mestu i prikazuje im najpovoljniju. U aplikaciji je implementirana „potrošačka korpa“ koja omogućava da se skupe svi proizvodi na jednom mestu, a zatim izvrši kupovina. Naravno, dodati proizvodi se mogu i izbaciti iz korpe ukoliko se kupac predomisli.

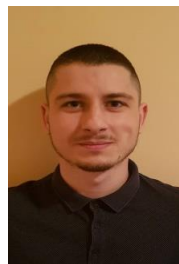
Dodavanje, izmena i brisanje prodavnica, proizvoda i kategorija u sistemu implementirano je tako da administrator mora ručno unositi informacije o njima. U daljem razvoju bi implementiranje servisa koji sam dobavlja ove informacije značajno poboljšalo i ubrzalo rad aplikacije. Takođe, izdvajanje administratora da sajt održava iz druge aplikacije bi bilo znatno sigurnije od eventualnih napada malicioznih lica. Korisnicima koji nisu registrovani je ipak omogućeno da vide najniže cene, što ih može zainteresovati i privući da se registruju i izvršavaju kupovinu preko sajta.

Logovanje je implementirano tako da korisnik može uneti ili **username** ili **email**, koji su jedinstveni u sistemu, što im omogućava da se prijave sa onim što lakše pamte. Takođe, u aplikaciji su implementirane opcije filtriranja proizvoda po kategorijama i pretraga proizvoda po nazivu što značajno ubrzava traženje željenog proizvoda, a dodate su i stranice po kojima su proizvodi raspoređeni radi bolje preglednosti sajta.

6. LITERATURA

- [1] Кончар Ј: „Електронска трговина“, Универзитет у Новом Саду, Економски факултет Суботица, Суботица, 2003
- [2] <https://www.eyerys.com/articles/types-e-commerce-models>
- [3] <https://www.c-sharpcorner.com/article/learn-about-angular-lifecycle-hooks>
- [4] <https://docs.microsoft.com/en-us/dotnet/core/about>
- [5] <https://docs.microsoft.com/en-us/ef/core/>

Kratka biografija:



Aleksandar Ljahović rođen je 19.05.1994. godine u Sremskoj Mitrovici. Završio je srednju ekonomsku školu 9. maj u Sremskoj Mitrovici 2013. godine. Fakultet tehničkih nauka u Novom Sadu je upisao 2013. godine. Ispunio je sve obaveze i položio je sve ispite predviđene studijskim programom.

CLOSURES IN JAVASCRIPT – ДЕТАЉНО ОБЈАШЊЕЊЕ И ПРИМЕНА**CLOSURES IN JAVASCRIPT – DETAILED EXPLANATION AND USAGE**

Стефана Томић, Факултет техничких наука, Нови Сад

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – У овом раду је представљена анализа затварања у JavaScript језику. Кроз примере је представљен JavaScript језик и могућност затварања као и објектни модел ДОМ са својим карактеристикама.

Кључне речи: Closures, JavaScript, ДОМ

Abstract – In this paper the analysis of closing in JavaScript language is presented. JavaScript language and possibility of closing is presented through examples as well as DOM object with its characteristics.

Keywords: Closures, JavaScript, DOM

1. УВОД

Примена затварања у JavaScript језику је широка. Од најједноставнијих примера скрипти које захтевају комуникацију корисника са одређеном HTML страном, до сложених примера употребе објеката у различитим наредбама [1]. Скрипт језици нису тако комплексни као уобичајени програмски језици, па нису подесни за писање комплексних програма. Углавном се користе када је потребно направити једноставне рутине којим се проверавају подаци на клијентској страни или када је потребно манипулисати елементима web странице. Ако је реч о сложенијим захтевима, онда се користи техника угњеждених објеката. У случају JavaScript-а код скрипта се или умеће директно у HTML документ или се из тог документа референцира неки JavaScript. У првом делу рада представљен је опис решаваног проблема где је описана употреба JavaScript-а на HTML странама, манипулација подацима у документима, употреба функције као и гранања. Описане су методе, приступ одређеним пољима, објектима као и prompt наредба. Такође описан је процес покретања функција у JavaScript језику који се користи у затварању.

2. ОПИС РЕШАВАНОГ ПРОБЛЕМА

Closure представљају функције које имају приступ променљивим вредностима које спадају у другу функцију скрипте. Овакав приступ је могуће извршити тако што се функција уграђује у другу функцију и на тај начин се изврши дефинисање ланца опсега.

Након тога затварање се може имплементирати и запамтити референцу из домена родитеља (из функције у којој има приступ променљивој). Такође променљиве које се користе у спољној функцији морају бити чуване у меморији како би биле доступне функцији затварања до краја процеса извршавања. Овакав приступ често може додатно успорити извршавање скрипте сходно превеликој употреби локалне меморије. Константним напредком JavaScript језика омогућено је побољшање употребе затварања. Битни фактори су да затварање памти променљиве из спољне функције иако је спољна функција затворена. Затварање памти референцу на коју је референцирана променљива функције. Такође је могуће извршавање редекларисање променљиве.

Истраживања из ове области показују да је сталним напредком script језика омогућена додатна употреба затварања и примена у решавању различитих комплексних задатака. Представљени су показни примери основних JavaScript функција на које је могуће применити затварање.

2.1. Употреба JavaScript основне функције у HTML-у

Функција се позива навођењем имена функције и прослеђивањем стварних параметара у заградама. Често је потребно да неку функционалност која се понавља у програму издвојимо као посебну целину која се може користити из остатка програма. Параметри функције су варијабле које се у функцију прослеђују приликом позивања функције и могу се користити у телу функције као обичне варијабле.

2.2. Уписивање текста у документ

InnerHTML property се користи за постављање садржаја HTML елемента. У следећем примеру Слика 1. JavaScript функција се ставља у делу HTML тага и извршава се када се кликне на дугме.

```
<script type="text/javascript">
  function izmenaTeksta()
  {
    document.getElementById('boldovano').innerHTML = 'Primer za master
rad';
  }
</script>
</head>
<body>
  <p>
    FTN: <b id="boldovano">br.2</b>
  </p>
  <input type="button" onclick="izmenaTeksta()" value="Klikni i izmeni
tekst" />
</body>
```

Слика 1. Уписивање текста у документ

НАПОМЕНА:

Овај рад проистекао је из рада чији ментор је био др Александар Купусинац, ван.проф.

2.2.1. Измена текста у документу

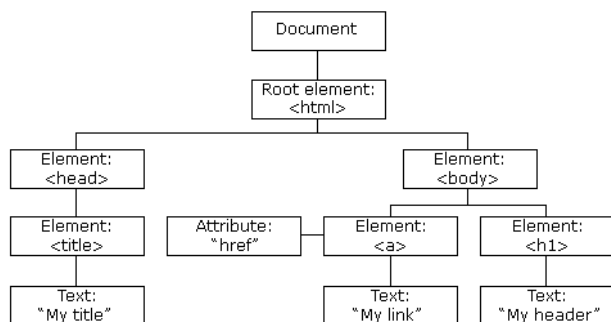
Предхони пример ће бити измењен тако што се мења текст који се исписује на почетку, текст дугмета као и текст након клика на дугме. Слика 2. представља другачији пример измене дугмета као и текст који мења почетни текст након клика на дугме.

```
<script>
function primerFunkcije() {
    document.getElementById("1111").innerHTML = "Tekst koji ce se ispisati nakon klika na dugme";
}
</script>
</head>
<body>
<h1>Master rad - Primer</h1>
<p id="1111">
    Tekst koji se ispisuje prilikom pokretanje stranice, a koji ce biti zamenjen nakon klika na dugme
</p>
<button type="button" onclick="primerFunkcije()">Probaj</button>
</body>
```

Слика 2. Измена текста у документу

2.3. Објектни модел HTML документа – DOM

Када се парсира HTML, он креира објектни модел HTML документа. Ово значи да он прави објекте који одговарају сваком HTML документу, атрибуту итд. Коришћењем ових објеката могуће је приступити вредностима HTML елемената. `document.getElementById()` је метода на објекту `document` која враћа референцу на објекат са датим идентификатором. Поља неких DOM објеката је могуће поставити на вредност неке функције што се генерално користи за реаговање на акцију корисника, тј. `event handling`. Начин на који се приступа и модификује садржај документа се назива Document Object Model или скраћено DOM. Објекти су организовани хијерархијски. Ова хијерархијска структура се односи на организацију објеката у web документу. Слика 3. представља једноставну хијерархију са DOM објектима.



Слика 3: Једноставна хијерархија DOM модела

- Window објекат– на врху хијерархије. То је спољни елемент објекта хијерархије.
- Document објекат– Сваки HTML документ који се прикаже у прозору читача постаје објекат Document. Document садржи садржај странице.
- Form објекат – Све што се налази између тагова `<form>...</form>` представља овај објекат.
- Контролни елементи форме (eng. Form control elements) – објекти форме садрже све елементе који су дефинисани у објекту као што су текст, поље за текст

(text field), дугме (button), радио дугме (radio button) и бокс за штиклирање (checkbox).

- Постоји неколико модела DOM-а:

- Legacy DOM– Ово је модел који се појавио током раних верзија JavaScript језика, тако да је добро подржан од стране свих претраживача, али дозвољава приступ само одређеним кључним деловима докумената, као што су форме, елементи форме и слике.

- W3C DOM – Овај модел документ објекта омогућава приступ и модификацију свих садржаја документа и стандардизован је од стране World Wide Web Consortium (W3C). Овај модел је подржан од стране скоро свих модерних веб читача.

- IE4 DOM - Уведен је у верзији 4 Microsoft-ovog Internet Explorer-a. IE 5 и новије верзије укључујући подршку за већину основних W3C DOM функција.

3. ОПИС КОРИШЋЕНИХ ТЕХНОЛОГИЈА И АЛАТА

JavaScript је платформски неутралан језик, као и HTML, што значи да би његов код (ако је писан по стандарду) требало да се извршава у оквиру читача клијента, без обзира која је хардверска машина или софтверско окружење у питању. Величина програма писаних у овом језику довољно је мала да може да се извршава и на машинама са лошијим перформансама. Још једна од предности JavaScript језика је његова интегрисаност са HTML-ом. У оквиру једне странице могуће је на произвољан начин комбиновати JavaScript и HTML код. Такође из JavaScript-а могуће је генерисати сам HTML код, у зависности од одређене акције корисника. Програмски код овог језика се може укључити у оквиру HTML странице: директним писањем кода у оквиру странице. `<SCRIPT LANGUAGE="JavaScript"> ...neki JavaScript kod... </SCRIPT>` Није неопходно да се наводи атрибут `language= "JavaScript"`, јер он има подразумевану вредност JavaScript.

3.1 JavaScript објекти

Постоји велики број већ дефинисаних објеката (document, window итд.) које конструише сам интерпретер, а програмер скрипте може дефинисати додатне објекте.

Објекти у JavaScript-у се дефинишу на следећи начин:

```
{ime_polja1:vrednost_polja2,
 ime_polja2:vrednost_polja2}
```

На пример, следећи објекат би могли користити да представимо корисника веб сајта: {email: "proba@gmail.com", ime: "Milan", prezime: "Markovic", indeks: 1111}.

3.1.1 Методе и приступ пољима објекта

За приступање пољима објекта користи се оператор „.“ (тачка). Током извршавања програма могуће је додавати нова поља објектима, читати и мењати њихову вредност. Пољима је могуће приступити и користећи синтаксу индексирања у низ. Следећи пример демонстрира приступ пољима објекта:

```
var o = {} //празан објект
```

JavaScript третира функције као грађане првог реда, што значи да је функције могуће третирати као било коју другу вредност: доделити их као вредност променљивој, прослеђивати у функције као параметар, враћати из функција као резултат итд.

3.1.2 Приступ пољима објекта из метода

Методе имају својство да припадају неком објекту. Као такве, могу приступити пољима тог објекта. За приступ пољима објекта којем метода припада, користи се специјална референца објекта `this`. У следећем примеру, користи се референца `this` за приступ пољима `name` и `lastName`.

```
var student = {name: "Dejan",  
  lastName: "Maksimovic",  
  sayHello: function() {  
    console.log(this.name);  
    console.log(this.lastName);  
  }  
};  
student.sayHello();
```

Други начин укључивања JavaScript-и је позив `js` документа. У оквиру тага се дефинише спољашњи документ у оквиру атрибута `src`. Спољашњи JavaScript код – омогућује коришћење/позивање исте скрипте у више HTML докумената `<script type="text/javascript" src="xxx.js"></script>`.

Није неопходно декларисати променљиву пре прве доделе вредности (аутоматски ће се извршити декларисање), али је и предекларисање дозвољено. Називи променљивих - имена променљивих могу да садрже бројеве и слова енглеске абетецеде, али први знак мора да буде слово енглеске абетецеде или симбол `"_"`, поред тога:

- Не могу се користити празна места у оквиру имена
- Не могу се користити резервисане речи као имена променљивих

Променљива или варијабла - описно речено је меморијска локација, "спремник" за одређену информацију/податак или вредност. Користи се за декларисање променљиве (декларација је креирање променљиве, а дефиниција значи и иницијализацију - постављање почетне вредности), чија синтакса је: `var imePromenljive`; Опционо могуће је извршити и њену иницијализацију.

3.1.3 Prompt наредба

Будући да JavaScript не обезбеђује једноставан метод за прихватање корисничких података, за ове потребе, користе се HTML форме и `prompt` дијалог прозори. `Prompt` дијалог прозор се појављује са једноставним пољем за текст (`textfield`). Након што корисник у `prompt` дијалог унесе текст, вредност се враћа. `Prompt dialog box` користи два аргумента: `string`, тј. текст који обично представља питање постављено кориснику опомињући корисника да одговори на питање и још један `string` који обично представља `default` одговор који је написан у пољу за унос. Ако корисник притисне ОК дугме, цео текст из дијалог прозора се враћа. Ако притисне `cancel` прозор биће му враћено `null`.

Синтакса је: `prompt("sometext","defaultvalue")`

4. ОПИС РЕШЕЊА ПРОБЛЕМА - ПРИМЕНА ЗАТВАРАЊА (CLOSURES)

Већина савремених програмских језика користи `lexical scoring` што подразумева извршавање функција уз помоћ варијабилног опсега који је дефинисан на почетку извршења а не у току позивања. Имплементација лексичког опсега захтева да интерно стање објеката JavaScript функције укључујује код функције и референцу на тренутни ланац опсега. Комбиновање објеката и опсега (опсега представља скуп променљивих) где су варијабле решене функције назива се затварање (`closure`) у компјутерској науци. Све JavaScript функције су затварања у случају да су објекти и да имају ланац опсега који је са њима повезан. Већина описаних функција се позива у истом ланцу обима који је коришћен када је функција дефинисана и у том случају није потребно затварање [4]. Најчешће се дешава да се угњеждени објект одређене функције врати из функције у којој је дефинисан. Кроз низ програмерских техника које укључују овај тип угњеждених функција затварања, примена угњеждених техника постала је честа у програмирању у JavaScript језику [6].

4.1 Имплементација затварања

Процес затварања (`closures`) једноставно је разумети и применити ако се користи правило лексичког опсега: функције се извршавају коришћењем ланца опсега који је примењен на почетку процеса дефинисања. Програмери у току имплементације сматрају да су затварања збуњујућа али се кроз детаљан опис имплементационог тока и ти проблеми решавају. Локална варијабла дефинисана у спољној функцији престаје да постоји када се спољна функција врати док се угњеждена функција може извршити користећи тренутни ланац опсега. Ако су локалне променљиве функције дефинисане у стеку процесора потребно је да престану да постоје у случају враћања функције [2].

Сваки пут када се јавља JavaScript функција врши се креирање новог објекта који обухвата локалне варијабле за тај позив а након тога се тај објект додаје у ланац обима. Када се функција враћа, објект који је везан за варијаблу се уклања из ланца оквира. У случају да не постоје угњеждене функције, референце на објект везивања не постоје и таложи се део програмског кода који се извршава [3]. Такође, уколико су угњеждене функције дефинисане, тада свака од тих функција има референцу на ланац обима а тај ланац обима се односи на објект везивања варијабле. Ако су угњеждени функционални објекти унутар спољашње функције у том случају се враћа или чува у неком својству а за то својство везана је и спољна референца.

`Closure` је способност функције да запамти и приступи свом лексичком `scope`-у када се функција извршава изван свог лексичког `scope`-а.

4.1.1 Примена `closure` функције

`Closures` функције су нам најпотребније када користимо `callback` механизам и креирамо функције које обрађују догађаје. Слика 4 представља пример који се често користи у пракси:

```

<script type="text/javascript">
function dodaj() {
    var blok = document.createElement("div");
    blok.innerHTML = "Kliknite na dugme.";
    document.body.appendChild(blok);

    var dugme = document.createElement("input");
    dugme.type = "button";
    dugme.value = "Dugme";
    dugme.onclick = klik; // definišemo event handler
    document.body.appendChild(dugme);

    function klik() {
        blok.innerHTML = "Kliknuto!";
    }
    //...
    dodaj();
}
</script>

```

Слика 4: Пример примене closure функције

Функција `dodaj()` креира један DIV блок и једно INPUT дугме. Да би се променила порука у блоку, потребно је кликнути на дугме и зато му задајемо `onClick` својство као референцу на функцију `klik()`.

Док корисник кликне на дугме, функција `dodaj()` је одавно завршена. Међутим, функција `klik()` која се тада позива, „одржава у животу“ све локалне променљиве њене надфункције и блок и дугме (иако јој дугме није потребно). Захваљујући томе, промениће текст у DIV блоку.

4.1.2. Closure памти променљиве из спољне функције чак иако је спољна функција извршена

Након коришћења - наредба `return` прекида извршење функције и постојање њених променљивих у привременој меморији.

Међутим „Closure“ памти променљиве из спољне функције чак иако је вратила неку вредност са `return`.

4.1.3. Closure памти референцу на коју указује променљива

Када се каже да „closure функција“ има приступ променљивој под тим се мисли да има приступ одређеном месту у меморији на коју указује та променљива тј. њеној референци у тренутку позивања „closure функције“. Уколико се на том месту у меморији тј. референци мења вредност, closure ће увек узети најновију тренутну вредност. У наредном примеру се види како унутрашња функција „памти“ референцу и користи тренутну вредност у меморији.

4.1.4. Редекларисање променљиве

Ако након дефинисања closure, променљивој која је потребна closure доделимо неку другу референцу (место у меморији које чува неку вредност), closure ће да користи референцу која је била у оптицају у тренутку дефинисања closure.

5. ЗАКЉУЧАК

Употребом затварања омогућено је:

- Додавање вредности променљивој коју враћа функција која се позива у скрипти
- Функција која се позива у скрипти извршава се само једном, постављањем бројача и враћањем функције у форми израза
- Променљива у том случају може приступити бројачу (counter) или надређеној функцији.

Функција у том случају има приватне променљиве и може приступити надређеном ланцу опсега и у случају да је надређена функција затворена.

Добра страна примене затварања је што се све може извршити на клијентској страни и знатно смањити коришћење меморије серверске стране.

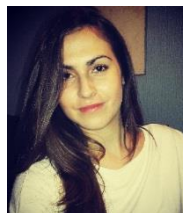
Sever – side scripting је Веб сервер технологија која омогућава да се кориснички захтев испуњава помоћу скрипта који се извршава на серверској страни како би се генерисале динамичке HTML странице. Применом затварања могуће је унапредити скрипту и знатно убрзати време извршавања и декларисања променљивих. Обично се користи да би се интерактивне Веб странице повезале са базама података, ради идентификације корисника, ажурирања садржаја, стварања разних дискусионих група и још због много тога. Разлика између скриптовања на серверској и корисничкој страни је у томе што се код корисничке стране скриптови извршавају у веб читачу корисника, најчешће помоћу JavaScript - а.

Client – side scripting технологије се обично односе на групу рачунарских програма на вебу који се извршавају на страни клијента, тј. од стране веб читача на клијентским рачунарима, уместо на страни сервера, од стране апликација веб сервера.

6. ЛИТЕРАТУРА

- [1] Computer Programming Skills & Maintenance, CHEN Yuanyi,ZHOU, JavaScript Closures Research & Typical Applications, 2011
- [2] David Flanagan, JavaScript - The Definitive Guide, 6th Edition, 2011
- [3] Daniel Jackson, functions, scope & closures
- [4] Advanced JavaScript Essentials, O'Reilly School of Technology's (OST)[6] Douglas Crockford, The Good Parts, 2012
- [5] Boris Fritscher, Zero to Hero JavaScript, 2012
- [6] Douglas Crockford, The Good Parts, 2012

Кратка биографија:



Стефана Томић рођена је у Сомбору 1992. године. Школске 2011/2012. године уписала се на Факултет Техничких Наука у Новом Саду. Дипломирала је на основним студијама 2016. године. Исте године уписала је мастер Примењено софтверско инжењерство.

RAZVOJ INFORMACIONOG SISTEMA ZA AUTO-ŠKOLE DRIVING SCHOOL INFORMATION SYSTEM DEVELOPMENT

Jelena Ljubišić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu predstavljena je perspektiva C# programskog jezika kroz tri paradigme – objektno-orijentisanu, funkcionalnu i konkurentnu. Svaka paradigma je analizirana prema svojim opštim odlikama, a zatim je analizirano na koji način su te karakteristike primenjene u C# programskom jeziku.

Ključne reči: C# programski jezik, Objektno-orijentisana paradigma, Funkcionalna paradigma, Konkurentna paradigma

Abstract – This thesis represents the perspective of C# programming language through investigation of three paradigms - object-oriented, functional and concurrent paradigm. Each of these paradigm is analyzed and their general characteristics are given. Furthermore, it is analyzed how are these characteristics defined in C# programming language

Keywords: C# programming language, Object-Oriented Paradigm, Functional Paradigm, Concurrent Paradigm

1. UVOD

Programska paradigma označava programski šablon ili način programiranja. Izučavanjem programske paradigme se upoznaju globalna svojstva jezika koji pripadaju toj paradigmi. Poznavanjem određene paradigme, olakšano se savlada svaki programski jezik koji toj paradigmi pripada.

Svaki programski jezik može podržati više paradigmi, što je i slučaj sa C# jezikom. On podržava i još neke paradigme pored onih koje su u fokusu ovog rada, međutim, izdvojene su tri najznačajnije i najzastupljenije paradigme, a to su objektna, funkcionalna i konkurentna.

C# programski jezik je pre svega objektno-orijentisan. Osnovni koncepti objektno-orijentisane paradigme – klase, objekti, svojstva, imenovani prostori, su neizostavni za druge dve paradigme, i koriste se u gotovo svim primerima koda koji su navedeni u radu.

2. OBJEKTNO-ORIJENTISANA PARADIGMA

Objektno-orijentisana paradigma je koncept programiranja koja uključuje kreiranje objekata koji modeluju poslovni sistem. Modeluju se zajedno operacije i podaci sa kojima operacija radi i čuvaju se kao jedinstvena komponenta.

Objektno-orijentisana paradigma se zasniva na konceptima apstrakcije, enkapsulacije, nasleđivanja (generalizacije) i polimorfizma.

Apstrakcija objedinjuje osnovne koncepte koje neki entitet obezbeđuje sa ciljem rešavanja nekog problema. Softverski model treba da bude pojednostavljen model realnog sistema, ali sa dovoljno podataka koji ga opisuju. Enkapsulacija je skup mehanizama koji obezbeđuje jezik za zaštitu od neželjenih izmena iz spoljnih izvora. U većini objektno-orijentisanih jezika, definisanjem privatnog pristupa promenljivoj u okviru klase se obezbeđuje da druge klase ne mogu da pristupe toj vrednosti direktno, već isključivo putem metoda za pristup i izmenu, što predstavlja skrivanje podataka. Enkapsulaciju karakteriše i skrivanje implementacionih detalja, upotrebom interfejsa. Preko interfejsa entitet nudi izvestan broj usluga i skriva implementacione detalje, koji za korisnika nisu bitni.

Nasleđivanje je mehanizam kreiranja novih od postojećih klasa. Izvedene klase imaju generalizovane karakteristike bazne klase, a pored njih i sopstvene specijalizovane karakteristike. To je važan princip objektno-orijentisane paradigme i naziva se generalizacijom i specijalizacijom.

Polimorfizam je kontekstno zavisno ponašanje, i usko je povezano sa konceptom nasleđivanja. Pored toga što izvedena klasa može da doda svoje specifično ponašanje, ona može i da izmeni ponašanje koje je nasledila od roditeljske klase kako bi njene specifičnosti bile uzete u obzir. U okviru polimorfizma su definisani pojmovi preopterećivanja i predefinisanja metoda ili operatora. Pored toga, postoji i parametarski polimorfizam, koji koristi upotrebu šablona metoda i klase, čime se omogućava višestruka upotreba.

2.1 Osnovni koncepti objektno-orijentisane paradigme

Klasa je softverski dizajn koji opisuje generalna svojstva predmeta modeliranja. Za definisanje klase se koristi ključna reč *class*, nakon čega se navodi naziv klase. Klasa opisuje tip objekta, dok objekat predstavlja upotrebljivu instancu (primerak) klase.

Polja i svojstva predstavljaju informacije koje objekti sadrže. Polja su poput promenljivih, jer se njihova vrednost može direktno čitati ili upisivati. Tradicionalne metode pristupa su u C# programskom jeziku zamenjene svojstvima, koji imaju efekat skrivenog modifikatora pristupa metoda. Standardna notacija za C# programski jezik je da se atribut obeležava nekim nazivom i piše se malim početnim slovom, a svojstvo se definiše nazivom istim kao i atribut i velikim početnim slovom.

To dozvoljava da se postavi vrednost atributa koristeći povezano svojstvo. Svojstvo omogućava da se koristi

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Aleksandar Kupusinać.

sintaksa karakterična za polja, a ne metode kada se pristupa privatnom polju.

Konstruktori su metode klase koje se pozivaju se automatski kada se kreira instanca neke klase. Koriste se da bi se inicijalizovale vrednosti atributa date klase. Konstruktori moraju imati isti naziv kao i klasa koju instanciraju. Podrazumeva se da konstruktor nema povratnu vrednost, i ključna reč *void* se ne navodi. Tipovi konstruktora u C# programskom jeziku su: ugrađeni konstruktor, konstruktor sa parametrima, konstruktor kopije, privatni i statički konstruktor.

Modifikator pristupa opisuje domen pristupa objekta i njegovih članova. Da bi se specificirala dostupnost objekta ili člana, u njihovoj deklaraciji se specificira neki od dostupnih modifikatora pristupa. U C# programskom jeziku su dostupni sledeći modifikatori: *private*, *public*, *protected*, *internal*, *protected internal* i *private protected*. Modifikatori pristupa održavaju enkapsulaciju uvodeći restrikcije kroz kontrolu pristupa.

2.2 Nasleđivanje u C# programskom jeziku

Višestruko nasleđivanje nije podržano u C# programskom jeziku. Svaka klasa može imati najviše jednu eksplicitno navedenu baznu klasu. Grupa klasa koje su povezane nasleđivanjem formiraju strukturu hijerarhije klasa. Posmatrajući klase kroz hijerarhiju, na vrhu je uvek klasa koja je direktan potomak ugrađene klase *Object*. Na taj način, C# obezbeđuje da je sve klase nasleđuju.

U C#-u, potklasa specificira nasleđivanje od natklase simbolom “:”, koji se navodi nakon navođenja naziva klase u njenoj definiciji. Nakon ovog simbola se navodi naziv klase koja se nasleđuje. Svaka klasa, bilo da je bazna ili izvedena, treba da enkapsulira svoju inicijalizaciju u konstruktoru. Svaka izvedena klasa treba da ima svoj konstruktor za specijalizovanu inicijalizaciju, ali često mora da pozove ponašanje baznog konstruktora.

Sve klase mogu biti nasleđene, osim ako se to eksplicitno ne specificira ključnom rečju *sealed*.

Apstraktne klase definišu svojstva i metode koje klase izvedene iz nje mogu da implementiraju. Ova klasa se može naslediti, ali se ne može instancirati. Klase koje se nalaze na vrhu hijerarhije su obično apstraktne klase. Može da sadrži potpunu implementaciju metoda koje definiše, može da definiše polja i konstante i deklaraciju konstruktora. Za specifikaciju se koristi ključna reč *abstract*. Značaj apstraktnih klasa je u održavanju generalizacije.

Interfejs je korisnički pogled na to šta neki entitet radi i razlikuje se od implementacije. Interfejs sadrži deklaraciju metoda, definiše dostupnost specificiranih operacija, bez implementacije istih. Za deklaraciju interfejsa koristi se ključna reč *interface*, nakon čega sledi ime interfejsa. Ne može sadržati promenljive, konstruktore ili kod sa implementacijom metode.

Klasa koja implementira dati interfejs treba da obezbedi implementaciju datih metoda u interfejsu. Svaka klasa može da implementira neograničen broj interfejsa. On enkapsulira znanje o objektu i predstavlja pogled sa korisničke strane. Implementacioni detalji entiteta se skrivaju interfejsom. Korisnik može da pristupi samo interfejsu i nema pristup podacima.

2.3 Polimorfizam u C# programskom jeziku

Ukoliko u baznoj i izvedenoj klasi postoje metode istog potpisa (isti naziv metode i isti ulazni parametri), onda je izvedena klasa zapravo predefinisala metodu iz bazne klase. Mogu se predefinisati samo metode koje su definisane u baznoj klasi kao virtuelne. U definiciji metode u baznoj klasi, pre definisanja povratnog tipa metode, dodaje se ključna reč *virtual*, čime se označava da je metoda virtuelna. Predefinisana metoda u izvedenoj klasi se obeležava u svojoj deklaraciji ključnom rečju *override*, pre nego što se navede povratni tip.

Preopterećivanje je pojam pridruživanja različitih operacija istom nazivu metode ili istom operatoru, a razlikuju se po prosleđenim parametrima. Odluka o tome koja se metoda poziva se donosi u fazi prevođenja, i to poređenjem tipova i broja argumenata.

Ideja predefinisanja i preopterećivanja metoda je da naziv metode nije više identifikator metode. Time se povećava fleksibilnost i robusnost softverskog sistema. Polimorfizam čini sistem otvorenijim za promene, jer se izmena sistema ovim olakšava.

Pored preopterećivanja i predefinisanja metoda, postoji parametarski polimorfizam ili generičko programiranje.

Generička metoda koristi parametrizovane tipove. Konkretni tip parametra se definiše kada se metoda poziva, i tada kompajler proverava da li je kod validan. Lista generičkih tipova podataka se nalazi u angular zagradi nakon navođenja naziva metode. Ukoliko postoji potreba za više generičkih tipova u metodi, odvajaju se zarezima. Po konvenciji pisanja C# koda se koristi veliko slovo “T” za definisanje generičkog parametra. Identifikator “T” može da predstavlja bilo koji tip podataka. Jedini uslov je da metoda mora da radi sa bilo kojim tipom podataka ili objektom koji mu je prosleđen.

3. FUNKCIONALNA PARADIGMA

Funkcionalna paradigma se zasniva na pojmu matematičkih funkcija. C# eksplicitno podržava neke koncepte funkcionalne paradigme, jer postoje brojne pogodnosti funkcionalnog programiranja: brže izvršavanje koda, čitljiviji kod, i pogodan je za kompleksna rešenja. Funkcija je ravnopravna sa ostalim tipovima podataka. Izraz u funkcionalnom programiranju je reprezentacija vrednosti, a ne sama vrednost. Izračunavanje izraza je neophodna operacija, i njegovo izračunavanje se svodi na jednostavne supstitucije i redukcije.

U funkcionalnoj paradigmi nije dozvoljena dodela koja može da izazove, promenu stanja definisane promenljive, odnosno bočni efekat. Svaki put se u naredbi dodele kreira nova promenljiva. S obzirom da u funkcionalnoj paradigmi ne može biti dodeljena vrednost promenljivoj nakon njene inicijalizacije, umesto petlji se za iteracije koriste rekurzivne funkcije. Za dodelu vrednosti funkcijskoj promenljivoj koristi se dodela imenovane metode, ili delegati.

3.1 Delegati

Delegat je tip podatka koji u C# programskom jeziku predstavljaju referencu na metodu. Koriste se za prosleđivanje metode kao argumenta drugim metodama. Ključna reč za inicijalizaciju delegata je *delegate*. Dostupni modifikatori pristupa za delegat su: *public*,

private, *internal* ili *protected*. Da bi se dodelila metoda delegatu, kreira se promenljiva delegatskog tipa koja ima potpis kompatibilan sa metodom koja mu se dodeljuje. Delegatski tip može koristiti generičke tipove kao parametre, čime se odlaže specifikacija jednog ili više tipova parametra ili povratnog tipa sve dok delegat nije inicijalizovan kao promenljiva.

U standardnim bibliotekama C# programskog jezika postoje ugrađeni delegatski tipovi. Takvi ugrađeni generički delegati su *Action* i *Func* delegati. *Action* ugrađeni delegat koji može da pokazuje samo na metode koje ne vraćaju parametre. *Func* delegat ima povratnu vrednost koja je generički predstavljena. Korišćenjem *Action* i *Func* delegata, kod je kraći, definicija delegata je lakša i brža.

Prava upotreba delegata je u izražavanju anonimnih funkcija koristeći lambda izraze.

U C# verziji 2.0 se javljaju anonimne metode, koje, obezbeđuju prečicu za kreiranje jednostavne i kratke semantike koja se upotrebljava samo jednom. Anonimne metode mogu biti korišćene i kao parametri druge metode.

C# se u verziji 3.0 proširuje sa lambda izrazima, kako bi kompletirali anonimne funkcije i obezbedili kratku notaciju koja je važna za njih. Lambda račun funkcije tretira kao izraze koji se postepeno transformišu do rešenja. U okviru lambda računa ne postoji koncept deklaracije promenljive i dodele naziva funkciji. Za transformacije se koriste redukcije i konverzije. Redukcije daju uputstva kako transformisati izraze iz početnog stanja u neko finalno stanje. Lambda račun naglašava pravila za transformaciju izraza i ne zamara se arhitekturom mašine koja to može da ostvari.

3.2 LINQ

Sa C# 3.0 se uvodi još jedna osobina za .NET framework, a to je LINQ (*Language Integrated Query*), koja omogućava kreiranje i izvršavanje upita na kolekcijama podataka koje implementiraju *IEnumerable<T>* interfejs. Generička klasa *IEnumerable<T>* ona predstavlja tip podataka za enumeraciju kolekcije podataka. To je interfejs tipa kolekcije, koji se primenjuje da bi se skupili svi podaci kolekcije koji su neophodni i da bi se enumerisali. Klasa *IEnumerable<T>* poseduje samo jedno svojstvo – *Current*, koje čuva elemente kolekcije za trenutnu poziciju enumeratora. Metode koje *IEnumerable<T>* klasa definiše su sledeće: *Reset*, *MoveNext* i *Dispose*.

LINQ upiti prepoznaju sekvence, koje predstavljaju objekte koji implementiraju *IEnumerable<T>*, i elemente, koji predstavljaju bilo koju stavku u sekvenci. Postoji više od pedeset operatora upita u *Enumerable* klasi uključenoj u *System.Linq* imenovanom prostoru.

LINQ implementira koncept odloženog izvršavanja (lenje evaluacije) kad se vrši upit nad podacima iz kolekcije, i neće izvršiti upit u konstruktorskom vremenu, nego u procesu enumeracije. Za enumeraciju se može koristiti *foreach* petlja, da bi se pozvala *MoveNext* komanda klase *IEnumerable<T>*, sa ciljem da se upit enumeriše. Lenja enumeracija je korisna kad je potrebno iterirati kroz beskonačnu petlju, i neće doći do prepunjavanja jer se *MoveNext* poziva samo kada je tražena. Čini da se kod izvršava brže jer kompajler ne mora da proveriti sve logičke izraze ukoliko jedan od njih da rezultat.

4. KONKURENTNA PARADIGMA

Konkurentnu paradigmu karakteriše više procesa koji se izvršavaju u istom vremenskom periodu, a imaju zajednički cilj. Za konkurentno programiranje potrebna je podrška u okviru programskog jezika ili biblioteka. Osnovni koncepti su nit, proces i zadatak.

Programska nit predstavlja osnovnu jedinicu operativnog sistema koja konkuriše za procesorsko vreme. Svaka nit ima sopstvenu stek memoriju na kojoj čuva kopije svojih lokalnih promenljivih kojoj samo ta nit može da pristupi, i svoj sigurnosni kontekst. Upotrebom niti se rešava problem propusnosti i odziva sistema.

C# programski jezik klasom *Thread* iz imenovanog prostora *System.Threading* obezbeđuje osnove za rad sa nitima, kao što je kreiranje i pokretanje niti. U C# programskom jeziku, svaki proces se pokreće izvršavanjem glavne programske niti koja se pokreće funkcijom *Main*. U toku izvršavanja procesa, moguće je kreirati proizvoljan broj niti iz bilo koje prethodno formirane niti.

Izvršna instanca svake aplikacije naziva se proces. Svaki proces poseduje resurse potrebne za njegovo izvršavanje. Procesi su međusobno u potpunosti izolovani, i za komunikaciju koriste međuprocensku komunikaciju. Niti jednog procesa dele hip memoriju i memorijski prostor, i mogu direktno da komuniciraju.

Nit se smatra blokiranom kada se njeno izvršavanje pauzira iz bilo kog razloga. Nit blokirana odmah predaje procesorsko vreme koje je zauzimala, i od tog momenta prestaje sa korišćenjem deljenih resursa dok ne bude zadovoljen uslov zbog kojeg je blokirana. Metode za blokiranje su *Join* i *Sleep*. Stanje niti definisano u svojstvu *ThreadState* ove klase za blokiranu nit je *WaitSleepJoin*.

4.1 Sinhronizacioni problemi

Za pozive funkcija koje su međusobno zavisne i dele resurse procesa, neophodna je sinhronizacija. Program u kom se izvršava više programskih niti istovremeno nije bezbedan ukoliko se neki objekat ponaša neočekivano u ovakvom okruženju.

Problem uslova takmičenja je situacija kada dve ili više niti istovremeno pristupaju deljenim resursima, pri čemu ishod izvršavanja programa zavisi od toga koja je nit prva pristupila određenom programskom bloku. Time su vrednosti deljenih promenljivih su nepredvidive i slučajne. Primer ovog problema je ažuriranje vrednosti iste promenljive od strane više niti. Vrednost promenljive koja je upisana od strane poslednje niti koja joj pristupa će obrisati prethodna upisivanja od strane drugih niti. Za dve niti koje istovremeno upisuju vrednost neke promenljive, dešava se situacija da se te dve niti takmiče čija vrednost će biti poslednja upisana u datu promenljivu.

Koncept napredovanja programa (engl. *liveness*) je karakteristika programa koji se sekvencijalno izvršava. Njime se garantuje da program može da se završi i da se stalno pravi određeni progres u programu. U konkurentnoj sredini sa deljenim resursima, postoji veća mogućnost da dođe do toga da program ne može da nastavi sa radom, što za posledicu može da ima da program nikada ne završi svoj rad.

Da bi se sinhronizacioni problemi izbegli, potrebno je obezbediti da pristup deljenim resursima tako da bude zaštićen od nesinhronizovanog pristupa i izmena od strane različitih niti. Osim upotrebe metoda koje blokiraju niti, mogu se implementirati i sinhronizacioni mehanizmi. Sinhronizacija u ovom slučaju obezbeđuje da za sekvencu instrukcija, koja se naziva kritičnom sekcijom, obezbedi da se sve instrukcije kritične sekcije izvrše bez prekidanja.

Za kontrolisanje pristupu deljenim resursima, koristi se zaključavanje (katanci i muteks mehanizmi) ili uzajamno isključivanje (monitori i semafori).

Određeni mehanizam se koristi u zavisnosti da li niti koje pristupaju deljenom bloku dolaze iz istog ili različitog procesa. Interne niti su niti kreirane u okviru istog procesa, dok su eksterne niti kreirane iz različitih procesa. Za sve navedene mehanizme, C# je obezbedio određene klase, koje definišu određene metode i svojstva kako bi obezbedile pouzdane sinhronizacione mehanizme.

4.2 Zadaci u asinhronom programiranju

Bazen objekata je šablon koji podrazumeva čuvanje inicijalizovanih objekata umesto da se objekti kreiraju i brišu na zahtev. Objekat se iz "bazena" uzima na zahtev, a nakon završetka rada nad tim objektom, umesto brisanja, objekat se vraća kako bi se mogao ponovo iskoristiti. *System.Threading.ThreadPool* je klasa C# programskog jezika koja implementira bazen za objekte tipa *Thread*. *ThreadPool* ima svojstvo *MaxThreads*, koje definiše maksimalan broj niti koje je moguće kreirati, i uobičajeno je da je broj niti manji nego što je broj zadataka koji čekaju na izvršavanje.

TaskQueue predstavlja red čekanja zadataka koji čekaju na izvršavanje redosledom kako su dodati u red i zauzimaju slobodne niti iz bazena niti. Nakon završetka izvršavanja zadatka, niti će biti dodeljen sledeći zadatak koji je na čekanju. Moguće je najviše jedan bazen na nivou procesa. U bazenu niti, metodom *QueueUserWorkItem()* se dodaje novi posao u red čekanja izvršavanja.

Šablon *Task-based asynchronous pattern* (dalje u tekstu skraćeno TAP) se koristi za proizvoljne asinhronne operacije u metodi i kombinuje status operacije i API koji je korišćen u interakciji sa ovim operatorima kako bi napravio jedinstven objekat. Ti objekti su *Task* i *Task<TResult>* tipovi, iz namespace-a *System.Threading.Tasks*. Objekat *Task* klase je centralna komponenta TAP šablona.

Task i *Task<TResult>* klase koriste niti koje se čuvaju u bazenu niti. *Task* klasa se koristi kada se metoda pokreće kao zadatak, a povratna vrednost nije potrebna. U suprotnom se koristi *Task<TResult>*.

Zadatak je jedinica programa koja može da se izvrši konkurentno sa drugim jedinicama istog programa. Zadaci ne moraju eksplicitno da budu pozvani. Kada program pokrene neki zadatak, ne mora uvek da čeka na njegovo izvršavanje pre nego što nastavi sa svojim. Kada se izvršavanje zadatka završi, kontrola ne mora da se vrati na mesto odakle je izvršavanje počelo. Svaki zadatak u programu može da bude podržan od strane jedne kontrolne niti ili procesa.

Pored upotrebe metode *StartNew*, koja kreira i pokreće zadatak, za kreiranje instance zadatka se najčešće koristi

statička metoda *Run*. Pokreće zadatak koristeći podrazumevane vrednosti i bez zahtevanja dodatnih parametara.

Ključne reči *async* i *await* C# programski jezik objavljuje sa verzijom 5.0, kao važan dodatak za asihrono programiranje u ovom programskom jeziku. Uvode se u sam programski jezik, da bi asinhrono programiranje bilo jednostavno za implementaciju i čitanje. *Async* ključna reč se dodaje ispred imena metode, a *await* se koristi u implementaciji metode, obično metode *Run* klase *Task*. Kada tok izvršavanja dođe do *await* dela, nit koja je pozvala metodu će iskočiti iz metode i nastaviti dalje sa izvršavanjem negde drugde. Asinhroni kod onda zauzima drugu, odvojenu nit. Sav tok izvršavanja nakon *await* ključne reči je planiran da se izvrši kada je zadatak završen.

5. ZAKLJUČAK

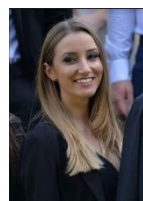
U uvodu rada je napomenuto da je C# programski jezik definisan prvenstveno kao objektno-orijentisan jezik u opštim definicijama. Kroz rad je pokazano da on može da podrži i druge dve navede paradigme - funkcionalnu i konkurentnu. Veliki broj opisanih pojmova u svojoj osnovi ima objektnu osobinu, bez obzira kojoj paradigmi pripadaju.

Ovim se omogućava da C# programski jezik u svojoj osnovi zadrži objektnu-orijentisanost, koja predstavlja odlično rešenje za modelovanje sistema koji odgovara principima ove paradigme, kako bi se izgradio softverski sistem koji je održiv i spreman na izmene. Konkurentnom paradigmom se omogućava izgradnja aplikacija koje imaju brz odziv, koje mogu da obavljaju više procesa istovremeno bez lošeg iskustva za korisnike zbog prestanka rada aplikacije. Funkcionalnim programiranjem u aplikaciji se odziv sistema ubrzava. Sa jedne strane, razumljivo je C# okarakterisati kao objektno-orijentisan jezik, ali on obuhvata još mnogo karakteristika van ove paradigme, koje su definisane u radu, i koje C# čine programskim jezikom pogodnim za razvoj raznovrsnih aplikacija.

6. LITERATURA

- [1] S.Kendall, *Object Oriented Programming using C#, first edition 2011*, Ventus Publishing ApS,
- [2] *Materijali sa vežbi nastavnog predmeta Sigurnost i bezbednost u smart grid sistemima*, master studija *Primenjenog softverskog inženjerstva* (pristupljeno u maju 2019.)
- [3] *Microsoft Developer Network*, <https://msdn.microsoft.com/en-us/library>
- [4] W.Anggoro, *Functional C#, third edition*, Packt Publishing, 2017

Kratka biografija:



Jelena Ljubišić rođena je u Novom Gradu 19.04.1993. Školske 2011/2012. je upisala Fakultet tehničkih nauka u Novom Sadu. Zvanje diplomirani inženjer elektrotehnike i računarstva stekla je 2016. godine. Položila je sve predmete predviđene planom i programom na master akademskim studijama na odseku: *Primenjeno softversko inženjerstvo*.

RAZVOJ MODELA ELEKTRIČNOG AUTOMOBILA U MATLAB SIMULINKU

DEVELOPMENT OF A ELECTRIC CAR MODEL IN MATLAB SIMULINK

Stefan Dupljanin, Marko Vekić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu prikazan je razvoj modela električnog automobila u okviru MATLAB Simulink-a. Model je razvijan korak po korak, gde je na početku automobil predstavljen kao crna kutija, delovanjem sile na tu kutiju uzrokuje se kretanje nekom brzinom. Takođe implementira se i model SUS motora, zatim model DC motora sa nezavisnom pobudom, strujni i brzinski regulatori kao i model slabljenja polja. Svakim korakom razvoja model postaje sve kompleksniji. Implementiranjem modela baterije i vozača završava se sa razvojem modela, jer su obuhvaćeni svi delovi električnog automobila. Na kraju su dati rezultati simulacije realne vožnje kako bi se proverila verodostojnost modela, gde su odzivi simulacije upoređeni sa očekivanim vrednostima, čime je dobijena potvrda ispravnosti modela.

Ključne reči: Električni automobil, MATLAB Simulink, SUS motor, DC motor sa nezavisnom pobudom

Abstract – The development of an electric car model within MATLAB Simulink is presented in this paper. The model is developed step by step, where at the beginning the car is presented as a black box, the action of force on that box causes movement at some speed. Also, the SUS engine model is implemented, then the DC motor with independent excitation, current and speed regulators as well as the model of field weakening. With every step of the development model becomes more and more complex. Implementing the battery and driver model ends with the development of the model, as all parts of an electric car are covered. In the end, the results of real-time simulation are given to check the credibility of the model, where the simulation responses are compared with the expected values, which gives a confirmation of the correctness of the model.

Keywords: Electric Car, MATLAB Simulink, Internal Combustion Motor, DC Motor with Independent Excitation

1. UVOD

Električna vozila sve više postaju deo svakodnevnice. Uzrok ovome je sve veća svest ljudi o tome koje sve prednosti donosi korišćenje električnog vozila umesto konvencionalnog.

Jedna od najznačajnijih prednosti jeste očuvanje životne sredine, odnosno smanjivanje emisije štetnih gasova, naročito u većim gradovima gde je zagađenost vazduha

sve veći problem. Posledica ove činjenice jeste ta da su vodeći svetski gradovi odlučili da do 2025. godine zabrane vožnju automobila u širem centru grada koji kao pogon koriste fosilna goriva (Pariz, Madrid, Atina, Meksiko Siti, Rim...) [1], što samo po sebi predstavlja podsticaj korišćenju električnih vozila. Dostupnost ovih vozila postaje sve veća razvojem automotiv industrije.

1.1. Zadatak rada

Zadatak ovog rada jeste da se napravi model električnog automobila, u MATLAB Simulinku, kako bi se posmatrale mehaničke pojave kao što su potrošnja baterije i pređeni put u zavisnosti od kapaciteta baterije. Sama realizacija je zamišljena tako da se napravi model običnog automobila sa motorom sa unutrašnjim sagorevanjem. Zatim će se takav model unapredi zamenom SUS (sa unutrašnjim sagorevanjem) motora električnim motorom sličnih karakteristika, kako bi se dobile iste ili slične performanse vozila u oba slučaja.

2. DINAMIKA KRETANJA VUČNOG VOZILA

Mehanički momenat se može predstaviti sledećom jednačinom:

$$M_m = -J_m \cdot \frac{d\omega_m}{dt} + M_{em} \quad (1.)$$

Kako je od interesa da u jednačini figuriše ugaona brzina izraz za ugaonu brzinu (2) će se uvrstiti u jednačinu (1) čime će se dobiti korigovana jednačina za mehanički momenat (3):

$$\omega_o = \frac{2 \cdot v}{D} = \frac{\omega_m}{N} \quad (2.)$$

$$M_m = -J_m \cdot N \cdot \frac{d\omega_o}{dt} + M_{em} \quad (3.)$$

Ukoliko se izraz 3. ubaci u jednačinu za dinamičku ravnotežu osovine točka 4. dobija se izraz 5:

$$J_o \cdot \frac{d\omega_o}{dt} = N \cdot M_m - \frac{D}{2} \cdot F_v \quad (4.)$$

$$(J_o + N^2 \cdot J_m) \cdot \frac{d\omega_o}{dt} = N \cdot M_{em} - \frac{D}{2} \cdot F_v \quad (5.)$$

Ukoliko se iz prethodne jednačine izrazi vučna sila F_v i ukoliko se uvrste izrazi 2. i 6. dobija se izraz 7 koji predstavlja jednačinu dinamike kretanja vučnog vozila:

$$m \cdot \frac{dv}{dt} = F_v - \sum F_{ot} \quad (6.)$$

$$m \cdot (1 + \varepsilon) \cdot \frac{dv}{dt} = \frac{2N}{D} \cdot M_{em} - \sum F_{ot} \quad (7.)$$

Veličina ε predstavlja koeficijent korekcije obrtnih masa i njegova vrednost se može dobiti izrazom 8:

$$\varepsilon = \left(\frac{2}{D}\right)^2 \cdot \frac{J_o}{m} + \left(\frac{2}{D}\right)^2 \cdot N^2 \cdot \frac{J_m}{m} \quad (8.)$$

Zbog implementacije modela u MATLAB-u izraz 7. je prilagođen tako da se kao rezultat modela dobije brzina vozila što je prikazano sledećom jednačinom:

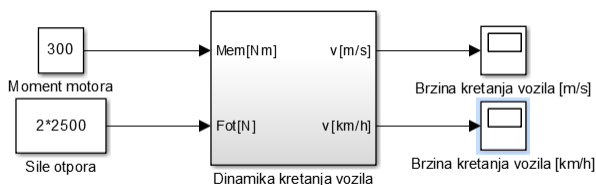
NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Marko Vekić, docent.

$$v = \int \left(\frac{2N}{D} \cdot M_{em} - \sum F_{ot} \right) \cdot \frac{1}{m} \cdot \frac{1}{(1+\varepsilon)} dt \quad (9.)$$

2.1. Predstava modela dinamike kretanja vozila u MATLAB Simulinku

Model je napravljen tako da kao ulazne parametre prima dve veličine: elektromagnetni moment (M_{em}) i sile otpora (F_{ot}). Na osnovu ulaznih parametara model proračunava brzinu koja se dobija na izlazu. Ono što je bitno napomenuti, izlazna brzina se može posmatrati u dve merne jedinice, [km/h] i [m/s]. Izgled modela u *MATLAB Simulinku* je prikazan na slici 1.



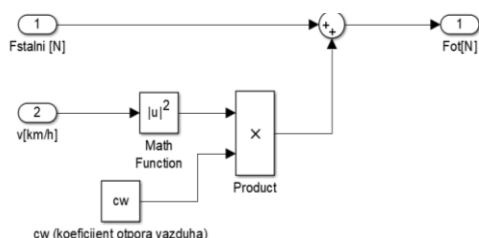
Slika 1 – Prikaz modela u MATLAB Simulinku

3. SILE OTPORA KRETANJU VUČNOG VOZILA

U najopštijem slučaju sile otpora koje deluju na vozilo u kretanju mogu se podeliti na unutrašnje i spoljašnje sile otpora.

Pod unutrašnjim silama otpora podrazumevaju se sve sile koje deluju pri prenosu snage od motora do točka, kako inercione tako i sile trenja elemenata transmisije [4].

Na slici 2 se može videti implementacija sile otpora kretanju u *Simulinku*, gde su otpori podeljeni na stalne i na otpore zavisne od brzine kretanja.



Slika 2 – Podsystem sile otpora kretanju

4. SUS MOTOR I TRANSMISIJA

4.1. SUS motori

Motoriga sa unutrašnjim sagorevanjem (SUS motori) su toplotni motori kod kojih produkti sagorevanja (dizel, benzin) svojim direktnim dejstvom vrše mehanički rad. Sam pojam motoriga sa unutrašnjim sagorevanjem obično se odnosi na one SUS motore kod kojih se proces sagorevanja obavlja s prekidima.

Osim ovih, postoje i SUS motori kod kojih se proces sagorevanja obavlja konstantno.

Predstavnicig ovih motoriga su gasne turbine, mlazni motori i većina raketnih motoriga [6].

Svakig SUS motorig u toku svog rada mora obaviti osnovna 4 procesa [7]:

- 1) Usisavanje,
- 2) Sabijanje (kompresija),
- 3) Sagorevanje, širenje (ekspanzija),
- 4) Izduvavanje.

4.2. Transmisija

Transmisija obuhvata sve delove i sklopove na vozilug koji prenose snagu od motoriga do pogonskih točkova vozila. Ona ima zadatak ne samo da prenese snagu od motoriga do pogonskih točkova vozila, nego i da izvrši promenu oblika pogonskih karakteristika motoriga, odnosno promenu parametara njegove snage [4]. Konceptigju, odnosno vrstu transmisigje najčešće određuju vrsta ugrađenog motoriga, a takođe i namena i kategorigja vozila, odnosno uslovi njegove eksploatacigje.

Podela transmisigje može da se vrši po različitim kriterijumigima, najčešćig je prema načinug prenošenja snage. Postoje četigri osnovne vrste:

- Mehaničke,
- Hidrauličke,
- Električne,
- Kombinovane.

4.3. Model SUS motoriga u MATLAB Simulinku

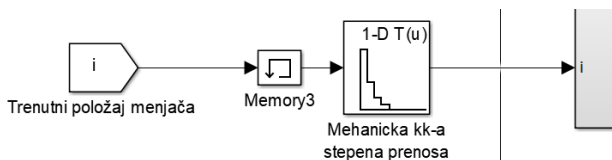
Za prikazivanje SUS motoriga u ovom modelu, korišćen je element *1-D Lookup Table* koji se nalazi u okviru *Simulink Library Browser*-a u biblioteci *Simulink* → *Lookup Table*. Samo modelovanje se realizuje tako što se zadaju četigri radne tačke, koje su linearno povezane formirajući krivug, čime se predstavlja zavisnost momenta motoriga od brzine obrtanja. Na slici 3 je prikazan izabrani motorig. Sa slike se uočavaju podacig od interesa, kao što su snaga i moment motoriga, i što je još bitnigje pri kojim brzinama se ostvaruju te vrednosti.

Mercedes Benz 190 E 1.8	
prenosi odnosi brzina pri 6000 o/min pri 4000 o/min pri 2500 o/min	
1.	3.910:1 50.4 km/h 33.6 km/h 21.0 km/h
2.	2.170:1 90.8 km/h 60.5 km/h 37.8 km/h
3.	1.370:1 143.8 km/h 95.7 km/h 59.9 km/h
4.	1.000:1 197.0 km/h 131.3 km/h 82.0 km/h
5.	0.810:1 243.3 km/h 162.2 km/h 101.3 km/h
R	4.270:1 46.1 km/h
Dif 3.460:1	
120 km/h V 2959 o/min	
130 km/h V 3206 o/min	
150 km/h V 3700 o/min	
0-100 km/h - 12.3 s	
max 185 km/h pri 4562 o/min	
pneu 185/65 R 15 O=1.894 m	
potrošnja	
90 km/h-6.7 l	
120 km/h-8.7 l	
grad -11.3 l	
težina 1150 kg nosivost 500 kg nosivost krova 100kg	
motor	
1797 ccm	
prečnik x hod klipa- 89.0 x 72.2 mm	
kompresija 9.1:1	
snaga 80kW (109 KS) pri 5500 o/min	
obrtni momenat 150 Nm pri 3700 o/min	

Slika 3 – Karakteristike izabranog motoriga

4.4. Model transmisije u MATLAB Simulinku

Model transmisije se može realizovati slično kao i SUS motor, dakle pomoću elementa *1-D Lookup Table*, gde se na osnovu ulaznog parametra trenutnog položaja menjača, na izlazu dobija vrednost ukupnog prenosnog odnosa u toj brzini, koji se dalje prosleđuje kao ulazni parametar podsistema *Dinamika kretanja vozila*. Na slici 4 prikazana je implementacija gore opisanog postupka.



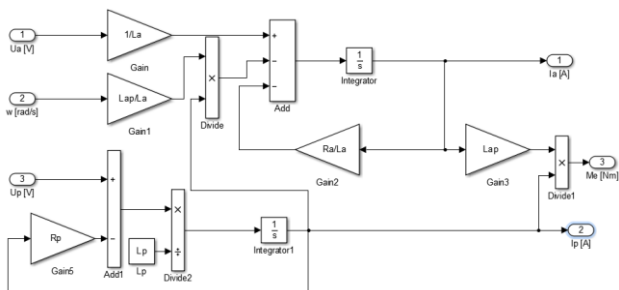
Slika 4 – Implementacija dobijanja stepena prenosa u modelu

5. MOTORI JEDNOSMERNE STRUJE

Motori jednosmerne struje spadaju u rotacione električne mašine kod kojih se vrši pretvaranje električne u mehaničku energiju rotacionog kretanja. Pripadaju dvopobudnim sistemima, jer imaju dva namota, statorski i rotorski [8].

5.1. Primena motora jednosmerne struje u okviru modela vozila

U okviru modela vučnog vozila predstava motora jednosmerne struje će se inkorporirati preko podsistema *Model električnog kola armature i pobude*, jer mehanički podsistem predstavlja samo vozilo. Na slici 5 je prikazana unutrašnjost podsistema *Model električnog kola armature i pobude*.



Slika 5 – Model električnog kola armature i pobude

6. UPRAVLJAČKI SISTEM MOTORA

6.1. Energetski pretvarač – čoper

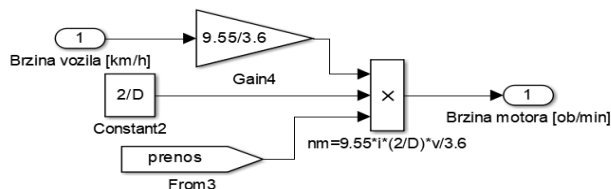
Čoperi ili impulsni regulatori jednosmernog napona su energetski pretvarači koji vrše pretvaranje jednosmernog napona konstantne vrednosti u jednosmerni napon promenljive srednje vrednosti, [4]. U radu se primenjuje četvorokvadrantni čoper.

Radom u sva četiri kvadranta se omogućava pogon u oba smera obrtanja, a reverziranje se može postići rekuperativnim kočenjem. Konceptija rada čopera, odnosno redosled uključivanja i isključivanja energetskih prekidača (tranzistora) može biti različit, u zavisnosti od željenog režima rada, da li kao jednokvadrantni, dvokvadrantni ili četvorokvadrantni.

Upravljanje može biti unipolarno ili bipolarno. U ovom modelu je odabrano unipolarno upravljanje.

7. SIMULACIJA VOŽNJE

Kako bi se došlo do mogućnosti simulacije vožnje potrebno je prilagoditi način zadavanja referentne brzine. Ovo u praktičnom smislu znači da je model potrebno prilagoditi da kao ulazni parametar prima brzinu vozila u [km/h]. Usled činjenice da regulator brzine kao ulazni parametar koristi brzinu motora u [rad/s] neophodno je implementirati konverziju koja će brzinu vozila pretvoriti u brzinu motora. Ovo je ostvareno blokom *Konverzija brzine vozila iz [km/h] u brzinu motora u [ob/min]* čija unutrašnjost je prikazana na slici 6.



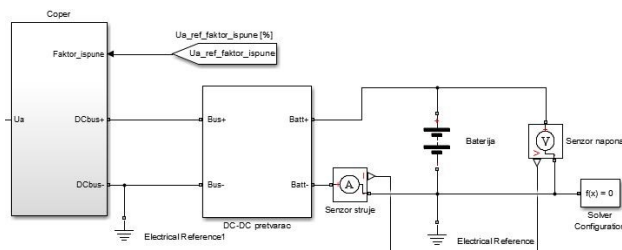
Slika 6 – Unutrašnjost bloka kojim se konvertuje brzina vozila iz [km/h] u brzinu motora u [ob/min]

8. BATERIJE ZA ELEKTRIČNA VOZILA

Baterije za električna vozila predstavljaju proizvod koji se koristi za napajanje pogona električnog vozila. Najčešće se koriste za napajanje viljuškara, vozila za golf terene, električne motocikle, električne automobile, kamione, kombije i druga električna vozila, dok sve veću primenu imaju i kod električnih biciklova.

8.1. Implementacija modela baterije u modelu

Model baterije je realizovan tako što je iskorišćen gotov model baterije koji se nalazi u okviru biblioteke *Simscape→SimPowerSystems→Sources→Battery*, slika 7.



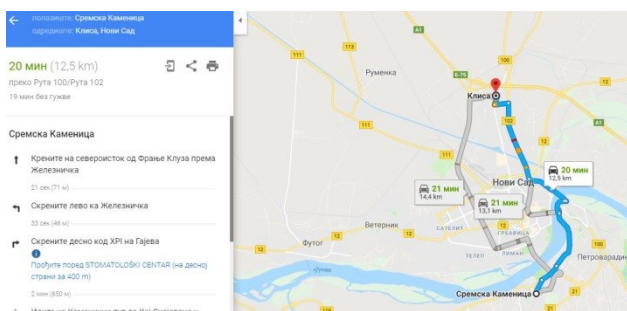
Slika 7 – Izgled baterije u okviru modela

Pomoću senzora struje i napona se detektuje struja i napon baterije, respektivno. Blok *PS-Simulink Converter* služi kako bi se fizički signal konvertovao u simulink signal. Ovo je neophodno iz razloga što biblioteka *SimPowerSystems* koristi fizički signal, dok se u ostatku modela koristi Simulink signal.

9. SIMULACIJA REALNE VOŽNJE

Kako bi se proverila verodostojnost modela, uzeće se parametri neke realne vožnje, u ovom slučaju ruta od Sremske Kamenice do Klise, gde će se predviđeno trajanje puta i pređena kilometraža proveriti sa odzivom simulacije. U zavisnosti od poklapanja ovih rezultata, oceneće se sama verodostojnost modela.

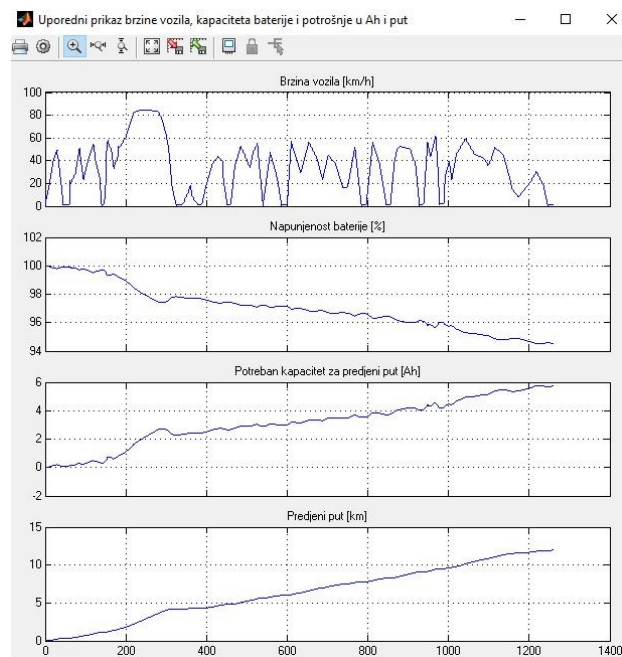
Na slici 8 se može videti izabrana ruta sa procenjenim parametrima, odnosno predviđenom kilometražom i vremenskim trajanjem puta.



Slika 8 – Izabrana ruta

Sa slike se uočavaju osnovni parametri ove rute, da je predviđeno trajanje puta 20 minuta, te da je razdaljina između početne i krajnje tačke 12,5 km.

Nakon zadavanja ulaznih parametara, brzine vozila u intervalu od 20 minuta, simulacija je pokrenuta. Na slici 9 se mogu videti rezultati simulacije, odnosno brzina vozila, napunjenost baterije, potrošeni kapacitet i pređeni put, respektivno. Uočava se da je pređeni put u simulaciji, slika 9, gotovo identičan predviđenom pređenom putu sa slike 8, iz čega proizilazi činjenica da model verodostojno oslikava realnu vožnju.



Slika 9 – Odziv modela

10. ZAKLJUČAK

Ovaj rad predstavlja osvrt na deo razvojnog procesa električnog automobila, jer je sastavni deo logičke

arhitekture sistema, gde se koristi modelovanje u *Simulinku* kako bi se opisala logika zahtevanih funkcionalnosti automobila.

U zavisnosti na kojim pojavama je fokus, model se može prilagoditi tako da se simuliraju željene veličine.

Kao što je na početku bilo naznačeno ovaj rad se bazirao na mehaničke pojave vezane za električan automobil, kao što su potrošnja baterije i pređeni put. U skladu sa interesnom sferom model se razvijao u pravcu posmatranja prethodno pomenutih stvari, ovo je razlog nekih zanemarenja u modelu,

11. IZVORI

- [1] N. Zimmermann, *Move is on to ban diesel cars from cities*, DW Germany, 2018.
- [2] <https://elbil.no/english/norwegian-ev-policy/>
- [3] https://en.wikipedia.org/wiki/Plug-in_electric_vehicles_in_Norway
- [4] B. Radojković, *Električna vuča*, Zavod za izdavanje udžbenika Socijalističke Republike Srbije Beograd, 1965.
- [5] I. Husain, *Electric and Hybrid Vehicles: Design Fundamentals*, Taylor & Francis e-Library, 2005.
- [6] J. Heywood, *Internal Combustion Engine Fundamentals*, McGraw-Hill Education, 1988.
- [7] C. F. Taylor, *The Internal-combustion Engine in Theory and Practice: Combustion, fuels, materials, design*, MIT Press, 1985.
- [8] K. Hameyer, *Electrical Machine I: Basics, Design, Function, Operation*, RWTH Aachen University Institute of Electrical Machines, 2001.
- [9] B. Jeftenić, Đ. Oros, V. Vasić, *REGULISANI ELEKTROMOTorni POGONI-REŠENI PROBLEMI SA ELEMENTIMA TEORIJE*, Akademska misao, 2004.
- [10] Siemens AG, *Catalog DA 12 – 2008 (DC motors, Sizes 160 to 630, 31.5 kW to 1610 kW)*, Siemens, 2008.
- [11] M.H. Westbrook, *The Electric Car: Development and Future of Battery, Hybrid and Fuel-cell Cars*, The Institution of Engineering and Technology, 2001.

Kratka biografija:



Stefan Dupljanin rođen je 1994. godine u Vrbasu. Gimnaziju „Isidora Sekulić“, završio je u Novom Sadu, 2013. godine. Fakultet tehničkih nauka, studijski program Energetika, elektronika i telekomunikacije upisao je školske 2013/2014 godine. Na studijama se opredelio za smer Elektroenergetika– Elektroenergetski sistemi i diplomirao 15.11.2017. god. Sledeće godine je upisao Master akademske studije, smer Energetska elektronika i električne mašine.



Marko Vekić je docent na Fakultetu tehničkih nauka u Novom Sadu, na Katedri za Energetsku elektroniku i pretvarače. Oblast interesovanja su energetska elektronika u prenosnim i distributivnim mrežama i kvalitet električne energije.

INSTANA ALAT ZA NADGLEDANJE PERFORMANSI APLIKACIJA I JAVA ATTACH API**INSTANA TOOL FOR MONITORING APPLICATION PERFORMANCE AND JAVA ATTACH API**Igor Milanović, Milan Vidaković, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj: U ovom radu je analiziran način funkcionisanja Instana alata za nadgledanje aplikacija i infrastrukture sa akcentom na Java attach API. Specificirani su zahtevi i dizajn, implementirana je aplikacija koja prikuplja i prikazuje parametre korisničkih aplikacija.

Abstract: This paper analyses the Instana tool for monitoring of applications and infrastructure with more details about Java attach API. It specifies the requirements and the design and describes the implementation of application which gathers client's application parameters and displays them.

Ključne reči: Instana, Java, agent, microservices, performance.

1. UVOD

U oblasti informacionih tehnologija i upravljanja sistemima, upravljanje performansama aplikacija (APM - *Application performance management*) predstavlja praćenje i upravljanje performansama i dostupnosti softverskih aplikacija. APM nastoji otkriti, dijagnostifikovati i predvideti složene probleme u performansama aplikacija kako bi održao očekivani nivo usluge. Naglim povećanjem potrebe za skalabilnošću sistema, pojavom i velikom popularnošću računarstva u oblaku (cloud computing) i distribuiranih servisa javlja se više nego ikada potreba za merenjem i nadgledanjem procesa unutar sistema.

U današnje vreme uopšte nije retkost da se aplikacija sastoji od dvocifrenog broja servisa koji međusobno ostvaruju interakciju i na taj način ispunjavaju zahteve korisnika. Jedna od najvećih kompanija koja koristi mikroservisnu arhitekturu je svakako Netflix, koja trenutno koristi oko 700 različitih vrsta mikroservisa [1].

Kada se ta brojka pomnoži sa koeficijentom skalabilnosti, lako se dolazi do cifre od nekoliko hiljada zasebnih servisa koji mogu da se nalaze u različitim ekosistemima. Glavni cilj APM sistema je povećanje kvaliteta iskustva krajnjeg korisnika (user experience) aplikacije koja se prati. Translacija prikupljenih metrika u vrednost od koje korisnik ima benefita je najvažniji zadatak svakog APM sistema [2].

NAPOMENA:

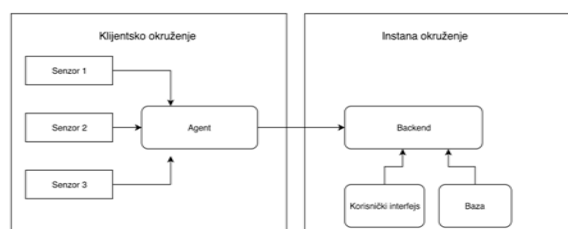
Ovaj rad je proistakao iz master rada čiji mentor je bio prof. dr Milan Vidaković.

2. PREGLED SISTEMA

Osnovni sistem se sastoji od: Senzora, Agent, *Backenda* i komponente za vizuelizaciju u okviru pretraživača koju ćemo nazvati UI (*User interface*, korisnički interfejs).

Senzori su komponente koje se izvršavaju u sklopu klijentske virtualne mašine tako što se infiltriraju u bajt kod u fazi izvršavanja aplikacije i prikupljaju samo promene metrika klijentskog koda koji presleđuje dalje agentu. Agent je komponenta koja je vezana za određenu virtualnu mašinu, odnosno programski jezik. Zbog jednostavnosti u daljem tekstu će biti opisivan agent za Java virtualnu mašinu. Odgovornost agenta je da detektuje da li na klijentskoj virtualnoj mašini postoji tehnologija za koju postoji odgovarajući senzor, i ako je uslov ispunjen, da se senzor instancira. Nakon što je proverio osnovni uslov, agent mora da prati životni vek senzora ne dozvoljavajući mu da prestane sa radom. Pored ove odgovornosti, agent mora biti u stalnoj vezi sa backendom da bi primenjivao razne promene u podešavanjima i da bi dobavljaio nove verzije izvršnih kodova senzora. Sve metrike koje stignu sa agenta na backend se moraju obraditi. Pod pojmom obrada podrazumeva se da li su klijentska pravila ispunjena, te stoga da li je potrebno da se aktivira određeni alarm. *Backend* pored pomenute funkcionalnosti takođe čuva metrike u bazi i time omogućava klijentu da ih pregleda.

Važno je napomenuti da klijent ima mogućnost da dodaje proizvoljna pravila koja su vezana za određenu tehnologiju. Na primer, pravilo može biti: ako je popunjenost memorije prešla devedeset procenata da se obavesti odgovorno lice. Određena pravila dolaze sa sistemom i njih korisnik ne može menjati; na taj način klijent dobija trenutnu vrednost od Instane bez potrebe da kreira svoja pravila. Na slici 1. je prikazana organizacija



Slika 1 – Prikaz organizacije komponenti

komponenti koje su objašnjene u tekstu iznad. Komponente su prikazane unutar okruženja u kome se izvršavaju. Važno je napomenuti da se sve komponente mogu izvršavati i samo u klijentskom okruženju ukoliko klijent želi takav pristup.

3. KORIŠĆENE TEHNOLOGIJE OSGI i Karaf

OSGi specifikacija opisuje modularni sistem i servisnu platformu za Java programski jezik koji implementira kompletan i dinamičan komponentni model što ne postoji u Java VM okruženjima. Aplikacije ili komponente, koje dolaze u obliku paketa, mogu se daljinski instalirati, pokrenuti, zaustaviti, ažurirati i deinstalirati bez potrebe za ponovnim pokretanjem. Upravljanje životnim ciklusom aplikacija se implementira putem API-ja koji omogućavaju daljinsko preuzimanje i upravljanje. Agent se pokreće pomoću Karafa.

Karaf je okruženje za izvršavanje aplikacija. Svaki senzor je opisan kao komponenta u Karafu. Karaf u toku izvršavanja odlučuje da li je potrebno da se aktivira određeni senzor u zavisnosti da li su ispunjeni unapred specificirani uslovi za pokretanjem senzora za određenu tehnologiju. Kada se uslov ispuni, Karaf preuzima paket za senzor sa dostupnog servera i pokušava njegovo pokretanje bez ometanja drugih komponenti. U praksi to znači da je interferencija između različitih senzora svedena na minimum. Kao što prati da li ima potrebe da se pokrene određeni senzor tako isto prati da li je potrebno ugasiti određeni senzor. Pored toga što se bavi životnim vekom senzora, Karaf obezbeđuje i sve zavisnosti koje su neophodne senzoru za pravilno funkcionisanje. Misli se pre svega na zahtevane biblioteke. Karaf takođe omogućava i verzionisanje senzora, što omogućava lako objavljivanje novih verzija senzora i takođe omogućava postojanje specifičnih senzora koji su zahtevani od određenih klijenata i vidljivi su samo njihovim okruženjima

Bajt kod instrumentacija

Počevši od Jave 5, JDK omogućava instrumentaciju bajt koda (bytecode). Java bajt kod je rezultat kompilacije Java programa, posrednog predstavljanja programa koji je nezavisan o hardveru na kome se izvršava. Ova tehnika ima za cilj mogućnost modifikacije bajt koda koji se učitava u JVM i izvršava na njemu - na primer, njegovo proširenje sa dodatnim instrukcijama ili druge promene izvornog bajt koda. Treba napomenuti da instrumentacija bajt koda ne uzrokuje promenu izvornih resursa, .class fajlova. On manipuliše bajtkodom on-the-fly (u letu, u tok izvršavanja koda) u vreme kada class loader pokušava da pristupi i učitava bajt kod odgovarajuće klase u JVM, proširujući ili zamenjujući bajt kod dobijen od originalnog resursa, sa svojom instrumentisanom verzijom.

Instrumentacijski interfejs pruža mogućnosti za dodavanje prilagođene implementacije transformatora koja će se aktivirati kada se bajt kod klase učitava u JVM, i može proširiti ili zameniti izvorni bajt kod klase pomoću prilagođenog.

Java agent

S obzirom da u realnim uslovima nemamo često mogućnost da restartujemo aplikaciju koju želimo da nadgledamo da bi se klase ponovno učitale i time omogućile da infiltriramo kod za instrumentaciju potrebno nam je drugačije rešenje. Koncept koji podrazumeva odvajanje koda za instrumentaciju i same aplikacije postoji u JVM

već neko vreme i poznat je kao Java agent. Java agent je aplikacija se obično isporučuje kao samostalna JAR datoteka (opciono sa dodatnim potrebnim zavisnostima), koja sadrži implementaciju instrumentacijske logike i može se povezati sa Java aplikacijama radi njihove instrumentacije. Postoje dva načina kako se Java agent može pokrenuti i učitati u instrumentovani JVM. To su statički i dinamički:

- Statičko pokretanje agenta tokom JVM pokretanja se postiže korišćenjem dodatnog JVM argumenta "-javaagent" i određivanjem lokacije JAR datoteke agenta kao vrednosti ovog argumenta (opciono, ako agent prihvati bilo koje parametre ili opcije, oni se takođe mogu proslediti kao deo argumenta): -javaagent: jarpath. Pomoću ovog pristupa može se učitati više agenata - potrebno je navesti nekoliko unosa argumenta "-javaagent", pri čemu se svaki od njih odnosi na pojedinačno učitano agenta. Ako se to uradi, agenti će biti učitan u sekvenci dok se odgovarajući argumenti "-javaagent" pojavljuju u JVM listi argumenata.

- Dinamičko pokretanje podrazumeva pokretanje agenta nakon pokretanja JVM i njegovo povezivanje sa nadgledanom aplikacijom. Ovo se postiže korišćenjem API Attach, koji je jedan od dijagnostičkih interfejsa koji su dostupni u modernim JVM. Ideja ovog pristupa je da pomoću API možemo da se povežemo sa JVM (priključimo se na određeni port, slično kao debug način) i učitamo agent iz određene JAR datoteke sa neophodnim neobaveznim argumentima, bukvalno u bilo koje vreme izvršavanja Java aplikacije

Byte buddy

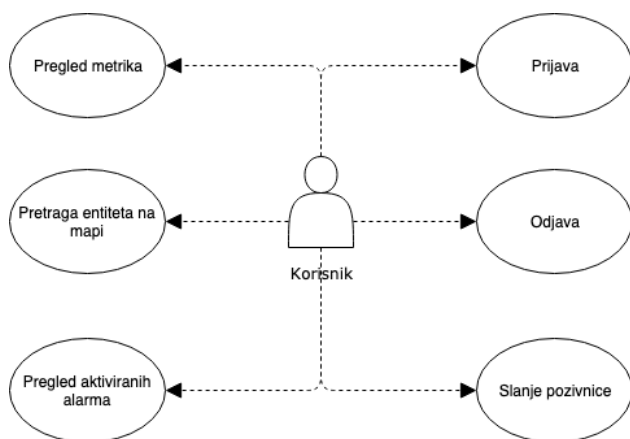
Java instrumentacija kakvu koristi Instana ne bi bila moguća bez alata poput Byte Buddy. Kao što smo videli na prethodnim primerima Instrumentation interfejs omogućava dodavanje transformatora čiji je osnovni zadatak transformisanje bajt koda učitanih klijentskih klasa. Ideja transformacije je urađena na niskom nivou, manipulacijom bajt koda što je i osnovni problem koji rešava Byte Buddy biblioteka. Najjednostavnije rečeno ona omogućava generisanje i modifikovanje Java koda u toku izvršavanja aplikacije i time omogućava programerima da jednostavnije, praktičnije i sa manje grešaka manipulišu kodom. Ova biblioteka se koristi u mnogo drugih biblioteka a jedna od najpoznatijih je Mockito u kojem omogućava pravljenje mokovanih objekata i njihovo podešavanje. S obzirom da biblioteka omogućava rukovanje bajt kodom na visokom nivou ona ipak nije onemogućila korisnicima da dodaju ručno delove bajt koda u aplikaciju

4. SPECIFIKACIJA SISTEMA

Zadatak obuhvata izradu aplikacije koja je zasnovana na Dropwizard okviru, koji se koristi kao osnova web aplikacije, za REST API i za čuvanje podataka u bazu, dok je front-end realizovan pomoću React okruženja. Korisnik može da se prijavi, odjavi, šalje pozivnice drugim korisnicima, pretraži metrike i entitete i pregleda aktivirane alarme. Ideja projekta je da omogući klijentu da ima bolju preglednost svog sistema.

Dijagram slučajeve korišćenja

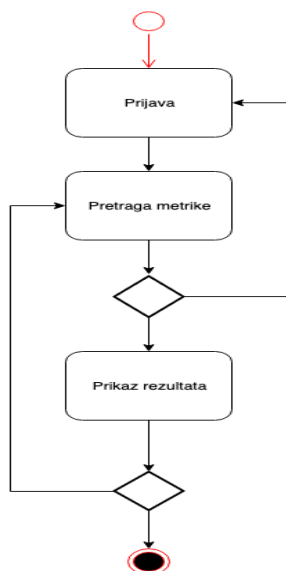
Dijagram slučajeve korišćenja predstavlja sve akcije koje korisnik može da izvrši i prikazan ja na slici 2.



Slika 2. Dijagram slučajeve korišćenja

Dijagram aktivnosti

Dijagrami aktivnosti su namenjeni modeliranju ponašanja programa. Oni prikazuju sekvencijalne i konkurentne korake u izvršavanju programa. Aktivnosti vezane za korišćenje aplikacije se nalaze na slici 3.



Slika 3. Dijagram pretrage metrika

Da bi se mogle koristiti funkcionalnosti aplikacije, korisnik mora biti prijavljen u sistem. Ukoliko pokuša da pristupi stranici za pretragu metrika biće preusmeren na stranicu za prijavu u sistem. Ukoliko je prijava uspešna korisnik se preusmerava nazad na željenu stranicu. Nakon uspešne prijave, korisnik može da pristupi stranicama za pretragu metrika, entiteta ili pregled alarma. Korisnik se može u bilo kom trenutku vratiti na stranicu za prijavu tako što se se odjaviti iz sistema.

5. IMPLEMENTACIJA

Projek je u osnovi Java web aplikacija. Dropwizard se koristi kao *back-end*, dakle za implementaciju REST API servisa, dok se za *front-end* koristi React klijentska aplikacija. Projekat je razvijen u *Intellij* razvojnom okruženju.

OSGI bundles

Instanci senzori su u stvari komponente koje su međusobno nezavisne i protokol komunikacije između njih je strogo definisan. Na taj način se sprečava eskaliranje zavisnosti i nameće labava sprega između komponenti. Labava sprega (eng. *loose coupling*) je pristup povezivanju komponenti u sistemu ili mreži tako da one zavise jedni od drugih u najmanjoj mogućoj meri. Spajanje se odnosi na stepen direktnog znanja koje jedan element ima o drugom.

Faze senzora

Svaki senzor se sastoji od dve faze *discovery* i *activation* (eng. aktivacija). Osnovna ideja *discovery* faze je da se provere uslovi koji su potrebni da se bi se aktivirao senzor. Najjednostavniji uslov koji treba da bude zadovoljen da bi se aktivirao Nginx senzor je da postoji proces koji je aktivan sa imenom "nginx: master".

Nakon detekcije takvog procesa parsira se lista argumenata sa kojom je pokrenut, interpretiraju se ti argumenti kako bi se dobila puna putanja do konfiguracionog fajla u kome se nalaze podešavanja za web server i sve to se prosleđuje Karafu kako bi on aktivirao Nginx senzor i zadao mu podatke koji su mu potrebni. Bitno je napomenuti da *discovery* faza se pokreće na svakih 30 sekundi, što omogućava klijentima da ako senzor nije bio pronađen u prethodnom ciklusu bude pronađen u sledećem. Karaf mora biti u mogućnosti da se štiti od pronalazaka koji traju predugo i postoji striktno definisan period vremena od kog *discovery* faza ne sme da bude duža. Karaf ima ograničen broj resursa sa kojima raspolaže i to je glavni razlog za ovako strogu politiku

Agent

Osnovna ideja agenta je da služi kao posrednik između senzora i *backenda* i da omogućava zajedničke funkcionalnosti sensorima. Kada se kaže na posredničku konekciju između *backenda* i senzora misli se da svi senzori komuniciraju isključivo sa agentom, potom agent dodaje autorizaciono zaglavlje na poruke i poruke prosleđuje dalje do komponenti *backenda*. Agent podatke sa senzora može da prosleđuje na više kanala. Kanali su: *event* (eng. događaj), *metrics* (eng. metrike) i *tracing* (eng. putanja izvršavanja). Svaki od kanala se drugačije obrađuje a i sama priroda podataka koji se šalju na ova 3 kanala su potpuno drugačiji. Konekcija između agenta i *backenda* je implementirana koristeći HTTP 2 protokol koji između ostalog omogućava postojanje stalne TCP konekcije kroz koju se šalju paketi. Agent šalje metrike na svaki sekund, kada bi se koristio starti HTTP protokol znatan overhead bi se koristio na uspostavljanje sigurne konekcije svake sekunde. Još jedna prednost novije verzije HTTP protokola je bidirekciona konekcija, u slučaju Instane to se iskorištava na način da backend može da šalje agentima komande koje će izvršiti lokalno i rezultat obrade poslati nazad. Sve to se izvršava preko jedne te iste konekcije

6. ZAKLJUČAK

Zadatak ovog rada bio je da se predstavi pregled Instanci sistema za nadgledanje aplikacija i infrastrukture, objasni njen značaj i da se pruži detaljniji pregled Javinog Attach API-ja koji omogućava da se Instancin Agent poveže sa korisničkom aplikacijom.

Glavna korist korišćenja Instane jeste ta da korisnik dobija bolji pregled ponašanja celokupnog sistema u realnim uslovima rada aplikacije i na taj način stvara prednost tako što može da vidi koji deo sistema je usko grlo ili ne funkcioniše u skladu sa dizjanom. Na taj način klijent zaokružuje ciklus kontinuiranog dostavljanja koda i njegovog nadzora.

Trenutna mana Instana sistema se krije u deljenom stanju pojedinih backend komponenti koje zbog toga ne mogu dovoljno brzo da skaliraju. Još jedna od mana je i ta da određene tehnologije za koje se rade senzori nemaju backward kompatibilnost što ima reperkusije i na senzore. Posledice su te da je potrebno napisati više senzora za istu tehnologiju u zavisnosti od konkretne verzije instalirane tehnologije.

Dalji razvoj Instana sistema je usmeren na omogućavanje bolje skalabilnosti backend komponenti i većoj stabilnosti senzor komponenti. Pored ova dva pravca značajni su i napori koji se ulažu u sigurnost sistema.

7. LITERATURA

- [1] Medium article, <https://medium.com/refraction-tech-everything/how-netflix-works-the-hugely-simplified-complex-stuff-that-happens-every-time-you-hit-play-3a40c9be254b>
- [2] APM fundamentals, <https://www.apmdigest.com/the-anatomy-of-apm-4-foundational-elements-to-a-successful-strategy>
- [3] Senonova teorema, https://en.wikipedia.org/wiki/Shannon%E2%80%93Hartley_theorem
- [4] Java 2 Platform, Enterprise Edition, <http://java.sun.com/j2ee/>
- [5] T. Schaeck, T. Hepper. Portal Standards. http://www.theserverside.com/articles/article.jsp?l=Portlet_API
- [6] Instana dokumentacija, <https://docs.instana.io/>
- [7] Designing Data-Intensive Applications, Martin Kleppman
- [8] Building microservices, Sam Newman

Kratka biografija:

Igor Milanović, rođen 20.04.1992, Rijeka, opština Rijeka, Hrvatska. Završio srednju tehničku školu „Ivan Sarić“ u Subotici. Nakon toga upisao se na Fakultet Tehničkih Nauka, odsek Računarstvo i Automatika.

Milan Vidaković je rođen u Novom Sadu 1971. godine. Na Fakultetu tehničkih nauka u Novom Sadu završio je doktorske studije 2003. godine. Na istom fakultetu je 2014. godine izabran za redovnog profesora iz oblasti Primenjene računarske nauke i informatika.

MODELOVANJE ENERGETSKIH PRETVARAČA U MIKROMREŽAMA**MODELING ENERGY CONVERTERS IN MICROGRIDS**Stevan Mihajlović, Vladimir Katić, *Fakultet tehničkih nauka, Novi Sad, Srbija*Octavian Cornea, *Politehnički univerzitet, Temišvar, Rumunija***Oblast – ENERGETIKA, ELEKTRONIKA I TELEKOMUNIKACIJE**

Kratak sadržaj – U ovom radu predstavljena su dva hibridna DC/DC pretvarača koji su našli primenu u mikromreži smeštenoj na Politehničkom univerzitetu u Temišvaru, naglašavajući njihov veliki prenosni odnos kao glavnu karakteristiku. Prikazani su analitički opisi, razmatranja dizajna, simulacija rada i eksperimentalni rezultati oba konvertora.

Ključne reči: Mikromreže, DC/DC pretvarači, Obnovljivi izvori električne energije

Abstract – This paper presents two hybrid DC/DC converters used in microgrid application at the UPT, highlighting large conversion ratio as its main feature. Analytical descriptions, design aspects, digital simulations and experimental results are presented.

Keywords: Microgrids, DC/DC converters, Renewable energy sources

1. UVOD

Mikromreže su delovi distributivnih mreža ili samostalne električne mreže manjih dimenzija koje karakteriše mogućnost samostalnog rada u određenim vremenskim intervalima na bazi primene distribuiranih generatora, uglavnom zasnovanih na obnovljivim izvorima energije [1]. Mikromreže nalaze svoju primenu u raznim aplikacijama kao što su napajanje udaljenih sela, ostrva, manjih naselja, bolnica, univerziteta, vojnih podsistema, poslovnih i stambenih zgrada, brodskih mreža, itd.

Po konvencionalnom shvatanju mikromreže su prosto proširenje tradicionalne AC distributivne mreže. Ipak, u poslednje vreme sve se više razmatra mogućnost primene DC napajanja, odnosno realizacija hibridnih distributivnih mreža sastavljenih od AC i DC podsistema [2]. DC sabirnice (i vodovi) na niskom i srednjem naponu, omogućuju direktno povezivanje obnovljivih izvora električne energije, što čini sistem jednostavnijim i efikasnijim. Umesto da su oni integrisani u AC mrežu preko dodatnog mrežnog invertora (DC/AC pretvarača), ovi resursi se u DC mrežu mogu povezati preko postojećih DC/DC pretvarača.

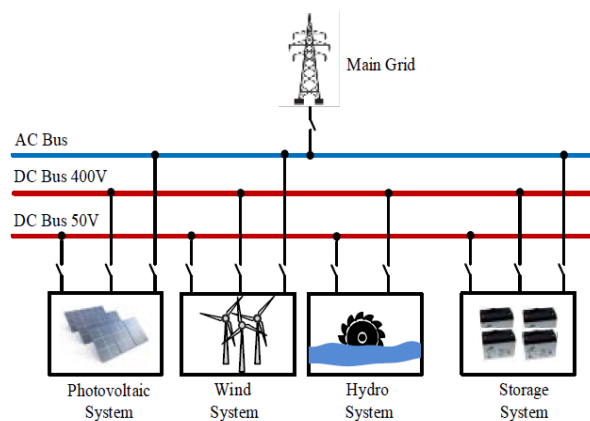
Međutim, zbog različitih snaga i priroda obnovljivih izvora, DC sabirnice (delovi DC mikromreže) mogu imati različite nominalne DC napone. Za povezivanje, upravljanje, normalan rad i protok energije u ovim

delovima mikromreže neophodni su odgovarajući unidirekcionni i bidirekcionni DC/DC pretvarači, koji mogu propuštati energiju u jednom ili oba smera.

U ovom radu su predstavljena i testirana dva tipa DC/DC pretvarača, koji su našli primenu u mikromreži „LAB D003“ smeštenoj na Politehničkom univerzitetu u Temišvaru [3].

2. MIKROMREŽA „LAB D003“

Topologija mikromreže „LAB D003“ prikazana je na slici 1 [3]. Sastoji se od četiri glavna sistema: fotonaponskog (FN), vetro, hidro i sistema za skladištenje energije. Energija se plasira do potrošača, superkondenzatora ili baterija preko dve DC sabirnice, nominalnih napona 50V i 400V sa mogućnošću povezivanja na standardni mrežni AC napon (400V, 50Hz). AC konekcija se koristi samo kada mikromreža radi sa distributivnom (glavnom) mrežom. Ceo sistem je potpuno konfigurabilan u smislu hardverske i softverske implementacije, te omogućava različite strukture mikromreže, koje mogu uključivati jedan, dva ili sve izvore. Ovo je moguće jer svaka komponenta sistema ima sopstvenu upravljačku strukturu, MPPT strategiju i elemente za skladištenje energije. Pozicije pretvarača i ostalih elemenata mreže mogu se videti na slici 2 [4].



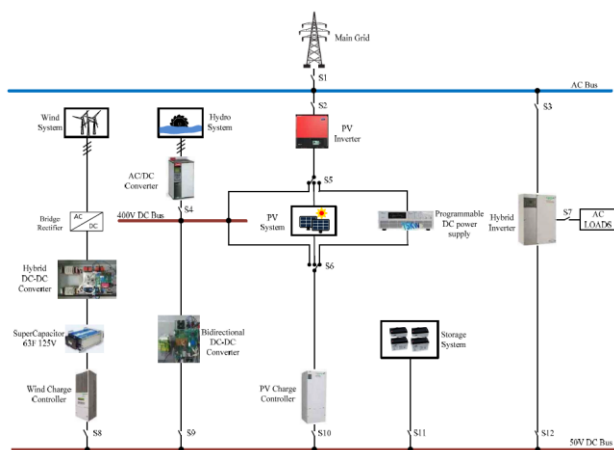
Slika 1. Topologija mikromreže [3]

2.1. Fotonaponski sistem

FN sistem je dizajniran za proučavanje ponašanja nizova panela povezanih sa mikromrežom za različite uslove i režime rada. FN sistem sastavljen je od nekoliko modula koji se mogu međusobno povezati na različite načine kako bi simulirali različite načine rada. Izvor se sastoji od 12 FN panela i programabilnog DC napajanja, koje može simulirati razne $I - V$ karakteristike.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Katić, red. prof.



Slika 2. Jedna od mogućih struktura mikromreže [3]

2.3. Vetro sistem

Vetro sistem koristi simulator vetroturbine (HIL emulator), napravljen pomoću AC DTC pogona (ABB ACS 800 sa asinhronim motorom) kojim upravlja kontroler (dSPACE interfejs ili Compact RIO iz NI) gde je turbinski model implementiran. Emulator pokreće sinhroni generator sa permanentnim magnetima. Sistem može raditi povezano s glavnom mrežom, bez skladištenja energije, kada je prekidač povezan sa stranom na kojoj se nalaze trofazni ispravljač sa prekonaponskom zaštitom. Ukoliko su priključeni skladišni elementi, generator daje energiju superkondenzatoru (Maxwell) preko hibridnog DC/DC konvertora u koji je uključen i mostni ispravljač. Energija koja se tako skladišti može puniti iste baterije preko kontrolera punjenja kao i FN paneli

2.4. Hidro sistem

Struktura hidrosistema mikromreže sastoji se od tri simulatora hidroturbine (HTS1, HTS2 i HTS3) sa tri tipa generatora (asinhroni, sinhroni sa DC eksitacijom i multifazni reluktantni sinhroni) sa istim AC DTC pogonom kao kod emulatora vetroturbina. Ovaj sistem ima tri AC/DC konvertora (Danfoss FC-302 invertori) sa modifikovanom kontrolnom tablom i vezom sa DC sabirnicama nominalnog napona 400V.

2.5 Bidirekcionni DC/DC pretvarač

Bidirekcionni DC/DC pretvarač povezuje DC sabirnice nominalnih napona 400V i 50V, tako stvara mogućnost transfera energije u oba smera. On omogućuje režim rada sličan spuštaču/podizaču napona, tj. ima mogućnost rada povezivanja višeg napona sa nižim izlaznim (*Step-Down* režim), odnosno podizanja napona (*Step-Up* režim). Za to postoje različita rešenja prikazana u literaturi, a rešenje primenjeno u mikromreži „LAB-D003“, je originalno i ima određene prednosti [4]. U daljem radu će biti prikazana dva DC/DC pretvarača iz pomenute mikromreže: 1. Hibridni unidirekcionni SIDC (eng. Switched-Inductor Step-Down Converter), i 2. Hibridni bidirekcionni pretvarač.

3. HIBRIDNI UNIDIREKCIONNI SIDC PRETVARAČ

Na slici 3 prikazano je električno kolo hibridnog unidirekcionnog SIDC pretvarača. On se primenjuje u sklopu

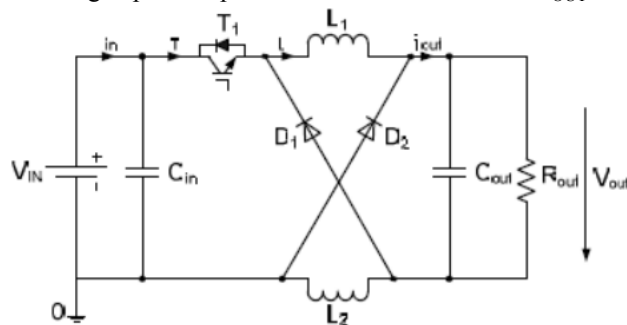
vetrogeneratora iza AC/DC pretvarača (slika 2). Moguća su dva stanja kola konvertora, kojima korespondiraju dva moguća uklopna stanja prekidačke komponente T_1 . Kada je T_1 zatvoren, dve prigušnice L_1 i L_2 su povezane na red i skladišti se energija u njima.

Tokom vremenskog intervala kada je T_1 otvoren, prigušnice L_1 i L_2 su povezane u paralelu. Tada se jedan deo ili celokupna energija skoncentrisana u magnetnom polju prigušnica predaje potrošaču. U zavisnosti da li je deo ili sva energija transferovana iz magnetnog polja prigušnica, konvertor će raditi u kontinualnom ili diskontinualnom strujnom režimu. Prenosni odnos konvertora dat je jednačinom:

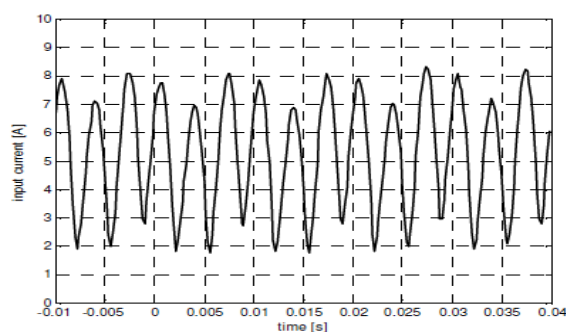
$$\frac{V_{out}}{V_{in}} = \frac{D}{2-D}, \quad D = \frac{t_{on}}{T} \quad (1)$$

Vidi se da ako se faktor ispune D kreće u opsegu 0-1, odnos izlaznog i ulaznog napona je od 0-0,5, što pokazuje da se radi o spuštaču napona.

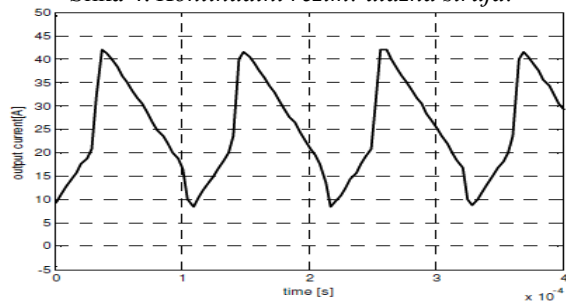
Kao primer rada ovog pretvarača uzet je slučaj da je ulazni DC napon setovan na $V_{IN}=180V$, a opterećenje iznosi $R_{OUT}=1,4\Omega$. Referentna srednja vrednost ulazne struje postavljena na $I_{IN}=5,5A$ daje izmeren izlazni napon od $V_{OUT}=36,6V$, koji se dobija za factor ispune $D=34,8\%$. Na slikama 4 i 5 dati su eksperimentalni rezultati merenja talasnih oblika struja na ulazu i izlazu pretvarača u kontinualnom režimu rada. Željena niska valovitost izlaznog napona se postiže filter kondenzatorom C_{OUT} .



Slika 3. Električno kolo konvertora



Slika 4. Kontinualni režim: ulazna struja.



Slika 5. Kontinualni režim: izlazna struja

4. HIBRIDNI BIDIREKSIONI PRETVARAČ

Električna šema hibridnog bidirekcionog pretvarača prikazana je na slici 6. Njegova topologija sastoji se iz tri glavna dela: 1. Podizača napona („boost“) bez izlaznog kondenzatora; 2. Komutacione kondenzatorske ćelije i 3. Izlaznog LC filtera.

Ulazni napon (V_1) obično je izvor nižeg napona, najčešće sastavljen od nekoliko baterija vezanih na red, pa je uloga prvog dela da ga poveća na vrednost izlaznog. V_2 je izvor višeg napona i u aplikaciji razmatranoj u ovom radu predstavlja sabirnicu napona 400V - 600V.

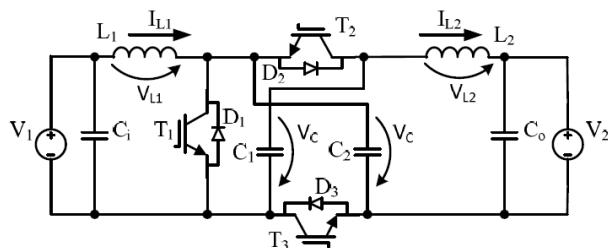
Kondenzatorska ćelija omogućava prenosni odnos napona konvertora. Oba naponska izvora imaju bidirekciono ponašanje tj. protok struje u oba smera.

Prekidačke komponente su takođe bidirekzione. U pitanju su IGBT ili MOSFET u paralelnoj vezi sa diodama koje omogućavanju protok struje u suprotnom smeru u bilo kom momentu, u zavisnosti od potreba sistema mikromreže. Prenosni odnos pretvarača, dat je relacijom:

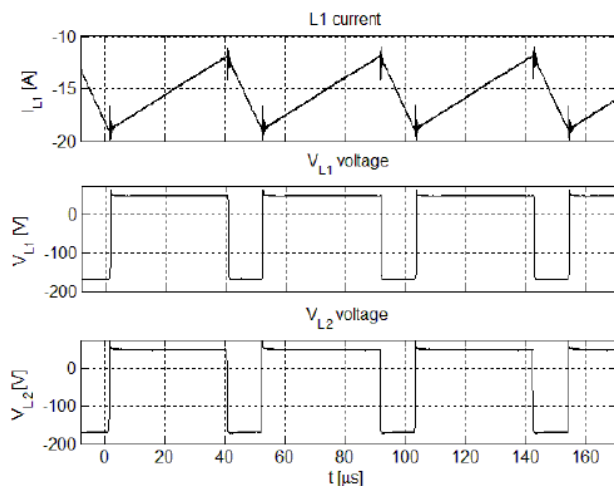
$$\frac{V_1}{V_2} = \frac{1-D}{1+D} = \frac{D'}{2-D'} \quad (2)$$

gde je D faktor ispuke T_1 , a D' faktor ispuke T_2 i T_3 . Kontrola rada prekidačkih komponenti je sledeća: jedan signal pokreće T_1 , a invertovan identični T_2 i T_3 .

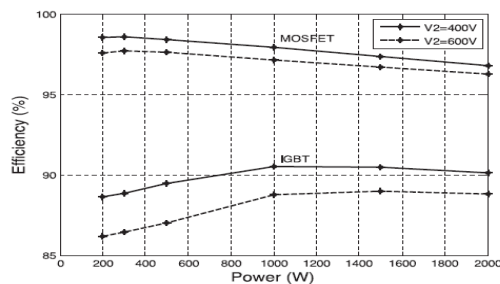
U zavisnosti od toga koja prekidačka komponenta je pokrenuta, drugačiji je smer struje koja protiče kroz prigušnice i konvertor je u „Step-Down“ ili „Step-Up“ režimu rada. Kao primer, na slici 7 prikazan je rad bidirekcionog pretvarača u „Step-Down“ režimu.



Slika 6. Električna šema bidirekcionog pretvarača.



Slika 7. Bidirekcionalni pretvarač kao spuštač napona



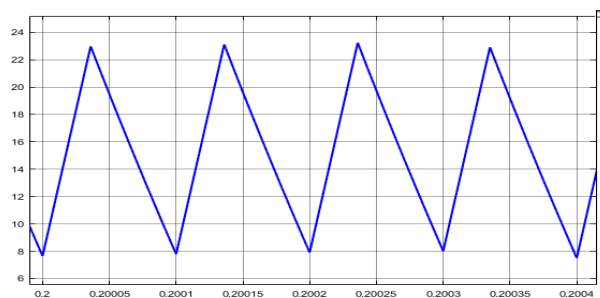
Slika 8. Procenjena efikasnost bidirekcionog konvertora za V_2 400V i 600V [4]

5. SIMULACIJA

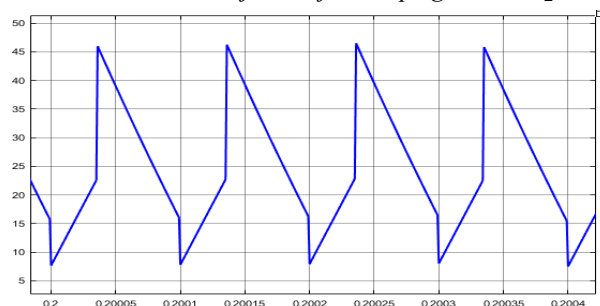
Modeli oba konvertora napravljeni su u programskom okruženju Matlab/Simulink. Energetski model konvertora i strujni kontroleri realizovani su kao posebni podsistemi. U nastavku su izdvojeni karakteristični talasni oblici rada oba konvertora za različite režime.

5.1. Hibridni pretvarač sa kalemima

Izvršene su simulacije za kontinualni strujni režim. Za simulaciju ovog režima ulazni DC napon iznosi 180V, a opterećenje na izlazu je $1,4\Omega$. Referentna vrednost struje na kontroleru je 5,5A. Za ovakvu postavku izlazni napon iznosi 36,6V približno. Faktor ispuke je približno 34,8%. Rezultati simulacija prikazani na slikama 9 i 10, u dobroj saglasnosti su sa stvarnim merenjima.



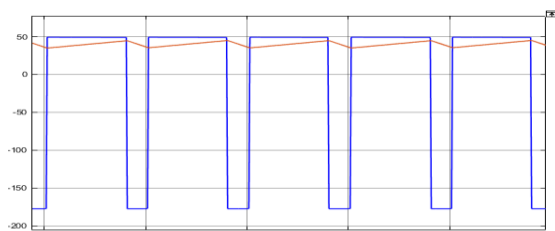
Slika 9. Simulacija: struja kroz prigušnicu L_2



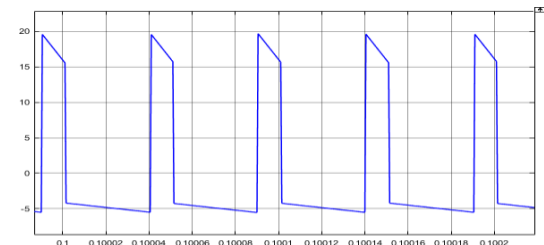
Slika 10. Simulacija: izlazna struja

5.2. Hibridni bidirekcionni pretvarač

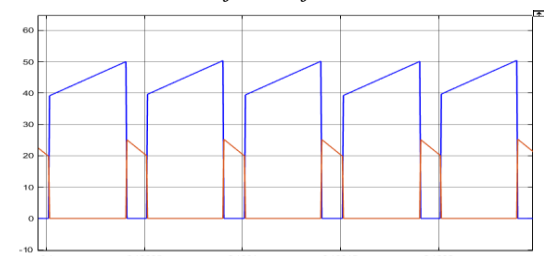
Za modelovanje bidirekcionog pretvarača naponi sabirnicama su stavljeni na 50V i 400V respektivno. Za odabran oblik referentne struje kontrolera, koja se kretala između 10A i 40A, na slikama 11-13 prikazani su uvećani talasni oblici. Na slici 11, prikazan je i talasni oblik napona na prigušnici L_1 . Poređenjem sa rezultatima merenja (slika 7) može se videti da postoji velika podudarnost. Ipak, da bi se to i kvantitativno pokazalo u narednom poglavlju je urađena verifikacija rezultata.



Slika 11. Simulacija: napon i struja prigušnice L_1



Slika 12. Simulacija: struja kroz kondenzatore



Slika 13. Simulacija: struja kroz T_1 i diode D_1 i D_2

6. VERIFIKACIJA REZULTATA

Upoređeni su rezultati simulacija sa stvarnim merenjima, a u tabelama 1 i 2 prikazana su odstupanja jednih u odnosu na druge. Kao parametri uzeti su relativna i srednja kvadratna greška. Najveća odstupanja zapažaju se kod talasnosti struja, koje su nešto veće nego kod stvarnih merenja. Na odstupanja u simulacijama oba modela uticalo je idealizovanje pasivnih elemenata (otpornosti, prigušnice i kondenzatori) kao i prekidačkih komponenti (IGBT, MOSFET i diode).

Kod modelovanja prvog konvertora na odstupanja je u velikoj meri uticala činjenica da je vetroturbina modelovana jednostavnim trofaznim naponskim generatorom, te za dalja unapređenja svakako bi trebalo staviti akcenat na verodostojnije modelovanje vetrogeneratora.

Modelovanje sabirnica u velikoj meri je takođe uticalo na odstupanja. Sabirnice su modelovane idealnim naponskim generatorima konstantnih napona što u velikoj meri utiče nepovoljno na verodostojnost simulacije. Kod modelovanja drugog konvertora uprošćenje modela strujnog kontrolera je sigurno u nekoj meri uticalo na verodostojnost rezultata simulacije.

Tabela 1. Odstupanja prvog konvertora

	$I_{t \max}$	$I_{L2 \min}$	$I_{L2 sr}$	$I_{L2 \max}$	$I_{out \min}$	$I_{out \max}$
I	20.59	8.97	16.22	22.58	8.58	41.32
II	20.32	8.69	16.03	22.21	8.82	42.09
III	20.29	8.64	15.41	22.33	8.65	41.68
Sim	22.5	7.87	15.8	23.1	7.94	45.3
ε	10.29	10.22	10.22	1.78	8.56	8.64
m	2.58	1.11	0.37	0.91	0.92	4.43

Tabela 2. Odstupanja drugog konvertora

	$I_{L1 \min}$	$I_{L1 sr}$	$I_{L1 \max}$	$I_{L2 \min}$	$I_{L2 sr}$	$I_{L2 \max}$
I	4.69	7.82	10.81	1.03	1.51	2.08
II	4.81	7.95	10.97	0.96	1.49	1.99
III	5.05	7.91	10.79	1.02	1.57	2.04
Sim	4.48	8.35	12.4	1.14	1.8	2.36
ε	7.63	5.79	14.22	13.62	18.16	15.88
m	0.49	0.56	1.89	0.17	0.34	0.40

7. ZAKLJUČAK

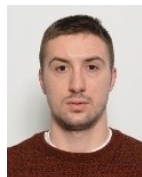
U ovom radu su opisana dva hibridna energetska DC/DC pretvarača. Digitalne simulacije rada modela stabilan rad konvertora u aplikacijama za koje su predviđeni. Talasni oblici dobijeni tokom laboratorijskih testova pokazuju dobro slaganje sa simulacionim.

Dalje modelovanje, osim poboljšanja prikazanih modela, u budućnosti bi išlo u pravcu integracije istih sa novoizgrađenim modelima ostalih delova mikromreže „LAB D003“ i stvaranjem kompletnog modela mikromreže koja bi služila za testiranje stabilnosti postojeće i novih konfiguracija.

8. LITERATURA

- [1] V. Mijailović, „Distribuirani izvori energije“, Akademski misao, Beograd, 2011.
- [2] T. Dragičević, X. Lu, J.C. Vasquez, J.M. Guerrero, „DC Microgrids - Part II: A Review of Power Architectures, Applications, and Standardization Issues“, IEEE Trans. on Power Electronics, Vol.31, No.5, May 2016.
- [3] C. Patrascu, N. Muntean, O. Cornea, A. Hedes: „Microgrid Laboratory for Educational and Research Purposes“, Int. Conf. on Environ. and Electrical Eng. (EEEIC), Florence (Italy), June 2016.
- [4] O. Cornea, E. Guran, N. Muntean, D. Hulea: „Bi-Directional Hybrid DC-DC Converter With Large Conversion Ratio for Microgrid DC Buses Interface“, IEEE Trans. on Industrial Electronics, Vol. 64, No.4, Apr.2017.

Kratka biografija:



Stevan Mihajlović rođen je u Rumi 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine odbranio je 2019. god. kontakt: stevan.db9@hotmail.rs



Vladimir A. Katić rođen je 1954. godine u Novom Sadu. Doktorirao je na Univerzitetu u Beogradu 1991. godine. Oblasni interesovanja su mu energetska elektronika, obnovljivi izvor električne energije, kvalitet električne energije električna vozila.



Octavian Cornea je diplomirao, magistrirao doktorirao na Politehničkom univerzitetu u Temišvaru 1996, 1997 i 2007. god. Oblasni interesovanja su mu modelovanje, simulacija upravljanje pretvarača energetske elektronike u raznim aplikacijama.

LOKALIZACIJA MOBILNIH UREĐAJA U ZATVORENOM PROSTORU POMOĆU MAŠINSKOG UČENJA**MACHINE LEARNING – BASED INDOOR LOCALIZATION FOR MOBILE DEVICES**Vukan Ninković, Dejan Vukobratović, Dejan Nemec, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – Usled velikog napretka u razvoju mobilnih uređaja, problem njihove lokalizacije u zatvorenom prostoru postaje jedna od najvažnijih tema današnjice. Najnovije ideje vezane za njegovo rešavanje su usko povezane sa principima mašinskog učenja. Mi predlažemo novi algoritam, koji posmatra vremensko – frekvencijsku strukturu signala nakon OFDM demodulacije kao sliku, i koristi sve benefite mašinskog učenja koji se već primenjuju u rešavanju problema obrade i klasifikacije slike. Na kraju vršimo poređenje različitih arhitektura po tačnosti klasifikacije i vremenu potrebnom za trening neuralne mreže.

Ključne reči: Lokalizacija u zatvorenom prostoru, mašinsko učenje, klasifikacija, preciznost, kompleksnost, 802.11ah

Abstract – With the proliferation of mobile devices, indoor localization has become an increasingly important problem. The latest ideas related to its solution are closely related to the principles of machine learning. We propose new algorithm which considers time – frequency representation of the signal, after OFDM demodulation, as a picture and uses all benefits of machine learning that are already applied in solving the problem of image processing and classification. At the end we compare different architectures by accuracy and complexity.

Keywords: Indoor localization, machine learning, classification, accuracy, complexity, 802.11ah

1. UVOD

Velikim napretkom u razvoju mobilnih uređaja i bežičnih (eng. wireless) tehnologija, servisi koji omogućavaju lokalizaciju u zatvorenim prostorima (npr. unutrašnjosti zgrade) postaju izuzetno važni. Za razliku od GPS (eng. Global Positioning System) sistema koji ostvaruje dobre performanse na otvorenom prostoru, kada postoji vidljivost između predajnika i prijemnika, lokalizacija uređaja u zatvorenom prostoru se susreće sa različitim preprekama i izazovima. Tu se pre svega misli na jako specifično radio okruženje u kom signal propagira, a koje uključuje različite efekte višestruke propagacije, senčenja, fedinga i kašnjenja. Istorijski gledano, postoje tri osnovna predloga za rešenje ovog problema [1]:

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dejan Vukobratović, red.prof.

- Ugao dolaska talasa na prijem (eng. Angle of Arrival - AoA) i pravac dolaska talasa na prijem (eng. Direction of Arrival - DoA)
- Vreme dolaska talasa (Time of Arrival - ToA), kao i vremenska razlika prispeća dva uzastopna talasa na prijem (eng. Time Difference of Arrival - TDoA)
- Pomoću snage primljenog signala (eng. Received Signal Strength - RSS). Ova tehnika je takođe poznata kao tehnika „otiska“.

Ekspanzijom mašinskog učenja, pojavljuju se autori koji pokušavaju da problem lokalizacije u zatvorenom prostoru reše upravo primenom principa mašinskog učenja, gde se neuralne mreže sve više nameću kao rešenje koje ima značajno bolje rezultate od gore navedenih tehnika. Kao ulaz u neuralnu mrežu, pomenuti autori su uglavnom koristili informacije o stanju kanala (eng. Channel State Information - CSI), tj. informacije koje predstavljaju uticaj kanala na svaki podnosioc OFDM (eng. Orthogonal Frequency-Division Multiplexing) sistema. Dati uticaj se ogleda u promeni amplitude i faze svakog podnosioca prilikom propagacije kroz specifično radio okruženje. Predloženo je nekoliko sistema koji koriste ove podatke, i koji pružaju bolje rezultate od klasičnih pristupa. Na primer, možemo navesti FIFS sistem koji koristi usrednjene amplitude CSI vrednosti [2], zatim DeepFi [3] i PhaseFi [4] koji koriste amplitude CSI vrednosti i podatke o fazi CSI, respektivno.

Takođe, postoje i hibridni modeli, koji pokušavaju da iskoriste kombinaciju starijih predloga (AoA, ToA) sa informacijom koju dobijamo pomoću CSI, i tu se izdvaja ResLoc algoritam [5].

Iako postoje značajna poboljšanja, rezultati koje postižu gore navedeni algoritmi još uvek nisu prihvatljivi. Zbog toga, naš pristup se prilično razlikuje od svih navedenih, kako po podacima koje koristimo u samoj neuralnoj mreži, tako i po činjenici da svi navedeni algoritmi pretpostavljaju da se određivanje lokacije vrši na samom mobilnom uređaju, što sa sobom nosi zahteve smanjene kompleksnosti i brzog izvršavanja. Naša pretpostavka je da bazna stanica određuje lokaciju, što značajno smanjuje potrošnju mobilnog uređaja. Takođe, na ulaz neuralne mreže dovodimo „sirove“, tj. neobrađene podatke sa fizičkog sloja, koje u daljoj obradi posmatramo kao sliku, i primenjujemo neuralne mreže koje su se već dokazale u problemima obrade i klasifikacije slike.

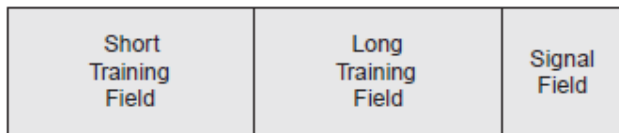
Poglavlje 2 opisuje model sistema (struktura ulaznih podataka i implementacija algoritma), poglavlje 3 govori o korišćenim arhitekturama neuralnih mreža, poglavlje 4 sadrži evaluaciju predloženog sistema, a poglavlje 5 predstavlja zaključak rada.

2. MODEL SISTEMA

2.1. Preambula 802.11ah paketa

Podaci korišćeni u ovom istraživanju su dobijeni simulacijom paketa specifičnog WiFi standarda 802.11ah (WiFi HaLow) u realnom okruženju (kancelarijski prostor). Ovaj standard je razvijen od strane IEEE tokom 2016. i 2017. godine, i osnovni cilj mu je da podrži širok spektar IoT (eng. Internet of Things) funkcionalnosti. WiFi HaLow funkcioniše u frekventijskom opsegu ispod 1GHz (oko 900 MHz), a njegov domet je dva puta veći u poređenju sa svim prethodnim WiFi standardima, tj. oko 1.5 km. Takođe, omogućava konektovanje velikog broja uređaja uz značajno redukovanu potrošnju energije. Naravno, povećanje dometa je uslovalo smanjenje brzine protoka, koja u ovom standardu iznosi od 150 kbps do 346 Mbps [6].

Svaki 802.11ah paket zauzima 2MHz i ima identičnu strukturu koja se sastoji od preambule i dela koji nosi podatke. Strukturu preambule možemo videti na slici 1:



Slika 1. Struktura 802.11ah preambule [6]

Tri osnovna polja svake preambule su (slika 1.) [6]:

- STF (eng. Short Training Field) koje se sastoji od 2 OFDM simbola, i služi za detekciju paketa i inicijalnu estimaciju frekvencijskog pomeraja.
- LTF (eng. Long Training Field) koje se takođe sastoji od 2 OFDM simbola i služi za finu estimaciju frekvencijskog pomeraja, ali i za procenu kanala na prijemu.
- SIG (eng. Signal Field) koje se sastoji od 2 OFDM simbola i sadrži informacije za konfiguraciju prijemnika (tip korišćene modulacije, tip kodovanja, količina prenesenih podataka).

Svaki OFDM simbol sadrži 56 korisnih podnosioca, pri čemu 52 podnosioca služe za prenos informacije od predajnika ka prijemniku (4 podnosioca služe kao piloti).

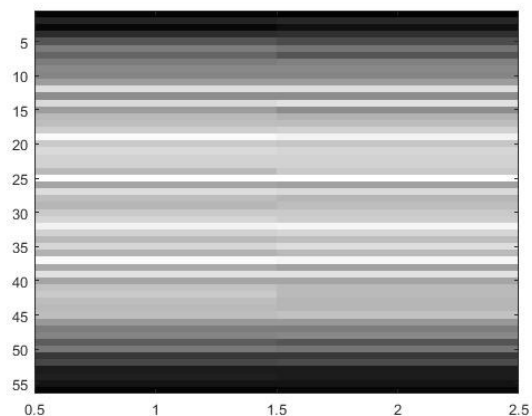
2.1. Struktura ulaznih podataka

Kako se ne bi zauzimao dodatni propusni opseg kanala (prenošenjem nezavisne informacije koja bi mogla da nam ukaže na lokaciju mobilnog uređaja), ideja je da se iskoristi preambula koja se prenosi u sklopu svakog 802.11ah paketa. U posmatranom scenariju mobilni uređaj (tj. predajnik) uvek šalje fiksnu preambulu, i na taj način dolazi do značajnog smanjenja kompleksnosti predajnika (kao i minimizovanja utrošene energije). Na putu od predajnika ka prijemniku svaka preambula će

doživeti različite distorzije (usled uslova propagacije), a što je prijemnik udaljeniji, slabljenje će biti veće.

Na strani prijemnika smo posmatrali LTF polje preambule, koje u suštini predstavlja matricu veličine 56x2 (posmatramo podnosioca koji nose informaciju i pilote). Ta matrica se dovodi na ulaz neuralne mreže, i dalje obrađuje kao slika, a vrednost jednog podnosioca u jednom OFDM simbolu predstavlja jedan piksel date „slike“. Naravno, s obzirom da su primljene vrednosti podnosioca kompleksne, na prijemu se prvo mora izračunati njihova amplituda, kako bi mogla biti primenjena odgovarajuća neuralna mreža.

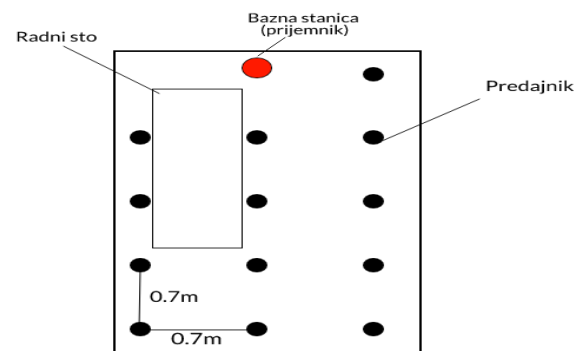
LTF polje jednog primljenog paketa možemo videti na slici 2:



Slika 2. LTF polje preambule primljenog paketa

2.1. Implementacija algoritma

Set podataka je formiran tako što smo sa svake od 13 različitih pozicija poslali po 1000 fiksnih preambula 802.11ah paketa, i na prijemu izdvajali LTF polje. Dato polje predstavlja ulaz u neuralnu mrežu, tj. neuralna mreža se trenira na slikama veličine 56x2. Merenja su izvršena u kancelarijskom prostoru veličine 5.5mx3,5m, pri čemu su i predajnik (mobilni uređaj) i prijemnik (bazna stanica) smešteni u istoj prostoriji. Bitno je napomenuti da je pozicija prijemnika fiksna, dok se predajnik pomera kao što je prikazano na slici 3. Razmak između susednih pozicija je bio 0.7m i po horizontali i po vertikali, a predajnik i prijemnik su bili na istoj visini, tako da smo problem posmatrali u dve dimenzije.



Slika 3. Položaji predajnika (crni krugovi) i prijemnika (crveni krug) unutar kancelarijskog prostora

Glavna pretpostavka na kojoj se bazira naš rad je da sa promenom pozicije predajnika dolazi do promene propagacionog okruženja, i samim tim do promene vrednosti amplitude primljene preambule. Sa svake pozicije mi smo imali jednu matricu veličine 56x2000 koju smo u preprocesiranju morali da „isečemo“ na 1000 slika veličine 56x2. Važno je reći da su se vrednosti amplitude LTF polja kretale u opsegu od 0.001 do 0.3, pa nije bilo potrebe za dodatnom normalizacijom podataka. Dobijene slike se čuvaju kao bin fajlovi, i ti fajlovi se dovode na ulaz neuralne mreže. Ovaj postupak se ponavlja za svaku poziciju.

3. ARHITEKTURA NEURALNE MREŽE

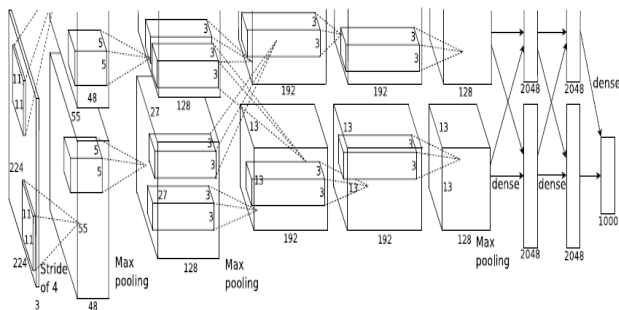
Jedan od najvažnijih parametara, koji ima značajan uticaj na performanse i kompleksnost sistema, predstavlja arhitektura neuralne mreže. Problem lokalizacije u zatvorenim prostorima, u našoj interpretaciji, može se posmatrati kao problem klasifikacije slike. Upravo zbog toga u datom istraživanju korišćene su dve arhitekture koje su pokazale zavidne rezultate prilikom rešavanja različitih vrsta problema u obradi i klasifikaciji slike. To su:

- AlexNet [7]
- ResNet [8]

Obe arhitekture predstavljaju konvolucione neuralne mreže, prilagođene konkretnom problemu i podacima kojima raspolažemo.

3.1. AlexNet

Originalnu AlexNet arhitekturu možemo videti na slici 4:

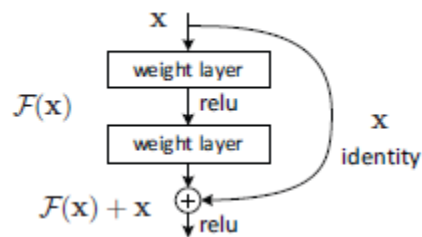


Slika 4. Originalna AlexNet arhitektura

Prvi konvolucionni sloj filtrira ulaznu sliku veličine 52x2 sa 32 kernela veličine 2x2x1, pri čemu je korak jedan piksel. Drugi konvolucionni sloj kao ulaz uzima izlaz prvog konvolucionnog sloja i filtrira ga sa 64 kernela veličine 2x2x32, korak je takođe jedan piksel. Treći konvolucionni sloj sadrži 128 kernela veličine 2x2x64, četvrti konvolucionni sloj ima 256 kernela veličine 2x2x128, a peti konvolucionni sloj ima 128 kernela veličine 1x1x256. Nakon konvolucionnih slojeva slede tri FC (eng. Fully Connected) sloja, pri čemu svaki FC sloj sadrži po 1024 neurona, a izlaz poslednjeg sloja predstavlja SOFTMAX aktivaciona funkcija veličine 13, koja nam prikazuje raspodelu verovatnoća po 13 različitih klasa (klasi sa najvećom verovatnoćom pripada data slika). U svim ostalim slojevima koristi se ReLu aktivaciona funkcija i korak je jedan piksel.

3.2. ResNet

ResNet arhitektura predstavlja arhitekturu koja omogućava lakše treniranje značajno dubljih neuralnih mreža, pomoću rezidualnog učenja. Osnovni blok rezidualnog učenja možemo videti na slici 5:



Slika 5. Osnovni blok rezidualnog učenja

Arhitektura mreže koju smo koristili je identična kao originalna ResNet arhitektura [8]. Jedina razlika je veličina filtra koja je svuda 2x2 (umesto 3x3). Aktivaciona funkcija je ReLu za sve slojeve sem poslednjeg (FC sloja), gde opet primenjujemo SOFTMAX funkciju.

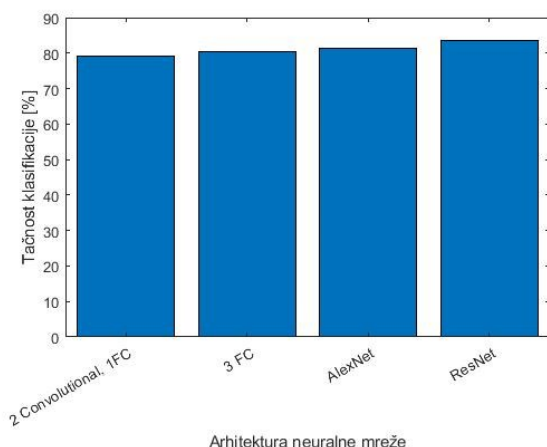
3.3. Arhitekture smanjene kompleksnosti

Predložene arhitekture imaju jedan nedostatak, a to je velika kompleksnost. Kako bi izbegli ovaj problem pokušali smo da napravimo što jednostavniji model, koji bi imao najbolji odnos performansi i utrošene energije. Ispostavilo se da arhitekture sa dva konvolucionna i jednim FC slojem ili sa tri FC sloja daju rezultate koji se u potpunosti mogu meriti sa rezultatima značajno kompleksnijih algoritama, a poređenje predloženih algoritama sledi u narednom poglavlju.

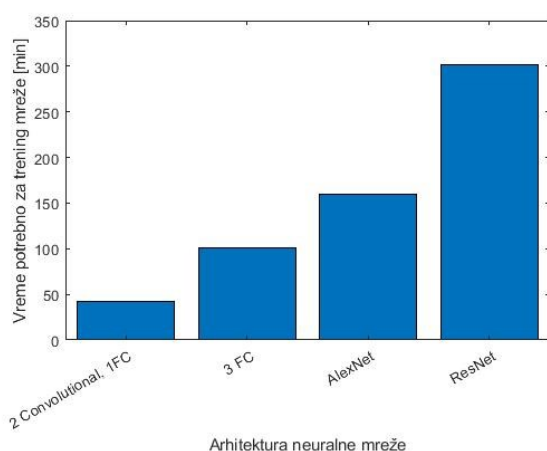
4. REZULTATI EKSPERIMENTA

Dobijeni skup podataka, koji se sastoji od 1000 slika veličine 56x2 za svaku od 13 klasa se na samom početku deli, tako da se 70% skupa koristi za trening, 20% skupa za validaciju i 10% skupa za test. Test skup nema nikakvih dodirnih tačaka sa procesom treninga, i on se koristi samo za proveru istrenirane mreže. Cilj je dobiti mrežu koja ima najbolji odnos performansi i kompleksnosti, pri čemu smo za kompleksnost mreže uzimali vreme koje je potrebno mreži da iskonvergira, tj. da završi proces treninga. Za sve eksperimente početna brzina učenja je bila 10^{-2} i data brzina se smanjivala za red veličine kada funkcija greške uđe u zasićenje. Treninzi različitih arhitektura neuralne mreže rađeni su pod identičnim uslovima (sa po 100 epoha) kako bi ih mogli uporediti po performansama i kompleksnosti, što možemo videti na slikama 6 i 7.

Sa slike 6 možemo videti da najbolju tačnost klasifikacije na test skupu od 100 slika ima ResNet arhitektura (83.7%) ali ova arhitektura ima ubedljivo najduži period treninga od čak 302 minuta (slika 7). Takođe, zbog dubine ove arhitekture, vrlo čest problem je preveliko prilagođavanje mreže trening skupu (eng. overfitting). Na drugom mestu i po preciznosti i po vremenu treninga se nalazi AlexNet (81.3% i 160 minuta). Treće mesto zauzima mreža koja se sastoji od 3 FC sloja sa preciznošću 80.5% i vremenom treninga od 101 minuta.



Slika 6. Tačnost klasifikacije različitih arhitektura neuralne mreže



Slika 7. Kompleksnost različitih arhitektura neuralne mreže (vreme potrebno za konvergenciju)

Na poslednjem mestu, ali sa ubedljivo najmanjim vremenom potrebnim za trening mreže nalazi se arhitektura koja se sastoji od dva konvoluciona i jednog FC sloja (79.2% i 42 minuta). Lako je uočljivo da se tačnost različitih arhitektura razlikuje za vrlo mali procenat, tj od 2% do 3%. Sa druge strane primetna je jako velika razlika u vremenu potrebnom za trening različitih arhitektura, koja u ekstremnom slučaju ide i do 260 minuta, tj. više od 4 sata. Jasno je da je kompleksnost mreže karakteristika o kojoj u praktičnim primenama moramo voditi najviše računa.

5. ZAKLJUČAK

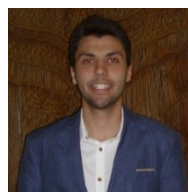
Primena principa mašinskog učenja za rešavanje problema komunikacionih sistemima predstavlja trend koji se u poslednjih godinu dana sve više razvija. Svakog dana pojavljuju se novi algoritmi koji postižu bolje performanse u odnosu na klasične i konvencionalne algoritme. U ovom radu smo predstavili potpuno nov pristup za rešavanje problema lokalizacije uređaja u zatvorenim prostorima, koji se oslanja na „sirove“ podatke sa fizičkog sloja, i ne zahteva prenos dodatne informacije od predajnika ka prijemniku. Takođe, dati algoritam koristi sve prednosti neuralnih mreža koje su otkrivene u rešavanju problema obrade i klasifikacije slike. Uporedili smo nekoliko klasičnih arhitektura

neuralnih mreža, i zaključili da postoji velika razlika u njihovoj kompleksnosti. Preciznost klasifikacije, koja je oko 80%, još uvek nije u potpunosti zadovoljavajuća, ali daje dobru osnovu za dalji rad i istraživanje ove teme.

6. LITERATURA

- [1] A. Khalajmehrabadi, N. Gatsis, D. Akopian “Modern WLAN Fingerprinting Indoor Positioning Methods and Deployment Challenges”, *IEEE Commun. Surv. Tutor.*, 2017.
- [2] J. Xiao, K. Wu, Y. Yi, L. Ni, “FIFS: Fine-grained indoor fingerprinting system”, *Proc. IEEE ICCCN'12*, pp. 1-7, August 2012.
- [3] X. Wang, L. Gao, S. Mao, S. Pandey, “CSI-based fingerprinting for indoor localization: A deep learning approach”, *IEEE Trans. Veh. Technol.*, vol. 66, no. 1, pp. 763-776, January 2017.
- [4] X. Wang, L. Gao, S. Mao, “CSI phase fingerprinting for indoor localization with a deep learning approach”, *IEEE Internet of Things J.*, vol. 3, no. 6, pp. 1113-1123, December 2016.
- [5] X. Wang, S. Mao, “ResLoc: Deep Residual Sharing Learning for Indoor Localization with CSI Tensors”, *Proc. IEEE PIMRC'17*, pp. 1-7, October 2017.
- [6] E. Perahia, R. Stacey, “Next Generation Wireless LANs: Throughput, Robustness, and Reliability in 802.11n”, Cambridge, 2008.
- [7] A. Krizhevsky, I. Sutskever, G. Hinton, “Imagenet classification with deep convolutional neural networks”, *Advances in neural information processing system*, pp. 1097-1105, 2012.
- [8] K. He, X. Zhang, S. Ren, J. Sun “Deep residual learning for image recognition”, *Proc. of CVPR*, pp. 770-778, 2016.

Kratka biografija:



Vukan Ninković rođen je u Šapcu 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Komunikacione tehnologije i obrada signala odbranio je 2019. god. kontakt: vukan.ninkovic@gmail.com



Dejan Vukobratović doktorirao je na Fakultetu tehničkih nauka 2008. god., a od 2019. je u zvanju redovni profesor. Oblast interesovanja je Teorija informacija i komunikacija.



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.

**UTICAJ PROPADA NAPONA NA RAD PUNIONICA ZA ELEKTRIČNA VOZILA
NAPAJANIH IZ SISTEMA KROVNIH FN ELEKTRANA****THE EFFECTS OF VOLTAGE DIPS ON ELECTRIC VEHICLES CHARGING STATIONS
POWERED BY ROOFTOP PV SYSTEMS**

Jovana Janković, Vladimir Katić, *Fakultet tehničkih nauka, Novi Sad, Srbija*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu iznet je predlog realizacije postavljanja javnih punjača za električne automobile na teritoriji Grbavice (Novog Sada). Navedeni su osnovni problemi sa kojima se ova industrija danas susreće. Opisan je električni automobil, opisana je postavka stavljanja FN elektrana, kao i njihova podela prema tipu i snazi. Prikazana je infrastruktura javnih punjača kao i predlozi postavke punjača na 7 lokacija. Dat je i proračun na osnovu kojeg je izvršen odabir opreme za napajanje i zaštitu, kao i simulacija kratkog spoja u DigSilent-u.

Ključne reči: Električni automobili, FN elektrane, punjači, pad napona, DigSilent

Abstract – This paper outlines a proposal of setting up of public chargers for electrical vehicles in the territory of Grbavica in Novi Sad is presented. The main problems that the electric vehicle industry is facing nowadays are introduced. The method of setting of rooftop PV system are represented in this paper as well as their division by type and power is described. The public charging infrastructure is presented and charger setting at 7 locations. There is also a budget based on which the power and protection equipment has been selected as well as simulation in DigSilent for short circuit.

Keywords: Electric cars, rooftop PV system, chargers, voltage drop, DigSilent

1. UVOD

Veliku emisiju štetnih gasova, a pogotovu gasova staklene bašte (GSB) izazvala je ubzanaindustrijalizacija, a naročito intenzivna motorizacija tokom XIX i XX veka. Danas se emisija štetnih gasova, koja potiče od vozila sa SUS motorom smatra jednim od najvećih ekološkim problema. Ovakav tip vozila predstavlja osnov modernog saobraćaja, a njihov broj i dalje brzo raste doprinoseći sve prisutnijim klimatskim promenama. U tom smislu, traže se adekvatna rešenja, koja će problem minimizovati ili potpuno eliminisati.

Jedna od mogućih alternativa jeste korišćenje automobila na električni pogon. Razvojem energetske elektronike i usavršavanjem baterija, upravljanje ovim automobilima i njihove performance postaju sve bolje.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Katić, red. prof.

Pored toga, energetska efikasnost električnih automobila je neuporedivo veća od automobila sa SUS motorima.

Za njihovo napajanje razvijeni su posebni uređaji koji služe za punjenje baterija automobila, tzv. punjači. Grupisanjem punjača razvijeni su novi infrastrukturni elementi, javne stanice za punjenje. Najčešće se baterije pune koristeći energiju uzetu iz distributivne električne mreže, prethodno prilagođenu upotrebom pretvarača energetske elektronike. Da bi bio eliminisan uticaj na emisiju GSB, razmatraju se opcije sa korišćenjem obnovljivih izvora. Međutim, tada je proces punjenja osetljiv na razne poremećaje, od kojih su kvarovi u mreži, odnosno nestabilnost napona, pojava propada napona i sl. najverovatniji.

U ovom radu je predloženo rešenje infrastrukture punionica za električna vozila zasnovano na krovnim fotonaponskim (FN) elektranama u delu Novog Sada, koji se zove Grbavica. Na demo primeru, koristeći napredan softverski alat DigSilent, razmatran je i uticaj kvarova u mreži na njihov rad.

2. ELEKTRIČNI AUTOMOBIL

Pogonski sklop električnog automobila čini električni sistem (baterija, elektromotor i uređaji energetske elektronike) i mehanički sistem, koji električnu energiju prenosi na točkove. Osnovne prednosti električnih automobila su veća energetska efikasnost, jednostavna konstrukcija pogonskog sklopa, nizak nivo buke, i jednostavno upravljanje motorom. Međutim, postoje i nedostaci, kao što su nešto lošije vozne karakteristike u smislu manjeg radijusa kretanja, relativno niže konačne razine, velika težina i gabariti baterija. Takođe, tu je i ekonomski elemenat, još uvek visoka cena.

Međutim, oni na indirektan način mogu doprineti većoj emisiji GSB, ako je električna energija, kojom se pune baterije, proizvedena u termoelektrani (TE) tj. iz uglja. U zavisnosti od tehnologije i efikasnosti rada, TE u atmosferu izbaci između 690 gCO₂/kWh_e i 990 gCO₂/kWh_e [1]. Električni automobili, koji se danas koriste, u proseku troše između 0,17 kWh/km i 0,2 kWh/km [2]. Ako se uzme u obzir da su gubici u prenosnoj mreži oko 10%, lako se dolazi do podatka da se prilikom vožnje potroši energija za čiju proizvodnju TE u atmosferu izbaci između 129 gCO₂/km i 218 gCO₂/km. Iako je to oko 10-15 puta manje nego što iznosi emisija GSB SUS vozila, moguća je dalja redukcija, ako bi se baterije električnih automobila punili energijom iz FN elektrana.

3. KROVNE FN ELEKTRANE NA GRBAVICI

Za napajanje električnih automobila iz krovnih FN elektrana razmotren je deo rada Novog Sada, Grbavica. Ovaj kvart karakteriše gusta naseljenost, veliki broj različitih stambenih zgrada i kuća, koje imaju različite oblike krovnih površina. Na slici 1 prikazan je satelitski snimak dela Grbavice sa jasno ucrtanim tipovima zgrada na osnovu krovne konstrukcije. Sa brojem 1 su označene zgrade sa kosim krovom, sa brojem 2 su označene kuće sa kosim krovom, a sa brojem 3 škole i zgrade sa ravnim krovovima. Kuće i zgrade sa kosim krovovima (1 i 2) razlikuju se jedni od drugih.



Slika 1. Određivanje tipskih krovova na Grbavici [3].

Na slici 2 prikazan je satelitski snimak kompletne Grbavice sa ucrtanim i pobrojanim različitim tipovima objekta na osnovu njihove krovne konstrukcije, za procenu krovne površine i njihovo grupisanje u tipске FN elektrane. Na bazi ovog pregleda mogu se postaviti (razlikovati) 4 tipa FN elektrana: 1. Na objekte sa kosim krovom sa solidnom orijentacijom jug-istok (crvena boja, ima ih oko 170); 2. Na objekte sa kosim krovom orijentisane većinom ka jugu (zelena boja, ima ih oko 64); 3. Na objekte sa ravnim krovom (plava boja, oko 17); 4. Na objekte sa ravnim krovom, koji imaju na sebi elemente koji prave senku kao što su antene, dimnjaci, gromobrani i dr. (žuta boja, oko 11) [3].



Slika 2. Pregled tipskih krovova na Grbavici [3].

Za elektranu tipa 1 prosečna dnevna proizvodnja električne energije 34,1 kWh, dok je prosečna mesečna proizvodnja električne energije 1.040 kWh. Ukupna godišnja proizvedena snaga iznosi 12.500 kWh [3].

Za potrebe tipa 2 može se zaključiti da je prosečna dnevna proizvodnja električne energije iznosi 10,8 kWh, prosečna mesečna 328 kWh, dok ukupna godišnja iznosi 3.930 kWh [3].

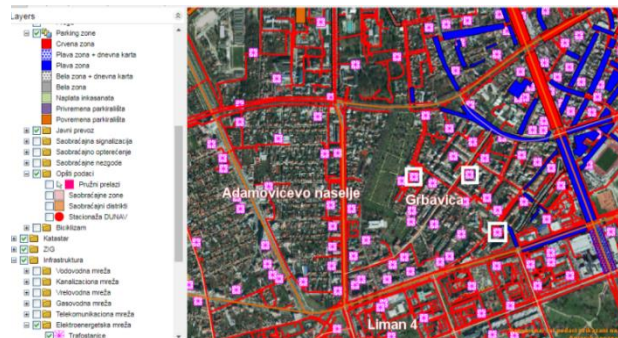
Za potrebe tipa 3 zaključeni su prosečna dnevna proizvodnja električne energije 94,3 kWh, dok je prosečna mesečna proizvodnja električne energije 2.870 kWh. Ukupna godišnja proizvedena snaga iznosi 34.400 kWh [3].

Za potrebe tipa 4 može se zaključiti da je prosečna dnevna proizvodnja električne energije 73,6 kWh, dok je prosečna mesečna proizvodnja električne energije 2.240 kWh. Na osnovu razmatranja došlo se do podatka da je na godišnjem nivou naša elektrana proizvela 26.900 kWh [3].

Izgradnjom FN elektrana na krovovima zgrada Grbavice pokrilo bi se oko 36% potrošnje električne energije. Ovo je okvirna vrednost, s obzirom na mogućnost određene greške zbog neophodnih pretpostavki i aproksimacija koje su učinjene. U najgorem slučaju, procena je da bi se obezbedila ušteda od oko 26%, što je odličan rezultat rasterećenja elektroenergetskog sistema.

4. POVEZIVANJE FN ELEKTRANA NA MREŽU

Dalje pitanje je mogućnost priključenja ovih FN elektrana na mrežu radi plasmana generisane energije. U saradnji sa inženjerima iz Elektrovojvodine, predložena su najpogodnija mesta za priključenje četiri tipa FN elektrana. Za prvi tip elektrane (zgrade sa kosim krovom) predviđeno je priključenje na novu TS u ulici Miše Dimitrijevića 9A. Za drugi tip elektrana (kuće sa kosim krovom) moguće je priključenje na TS u ulici Doža Đerđa 50, za treći tip elektrane (zgrade sa ravnim krovovima) na TS u ulici Alekse Šantić 27, kao i četvrti tip elektrane. Na slici 3 su označene (uokvirene belim) opisane TS.



Slika 3. Prikaz TS za FN elektrane na Grbavici [3].

5. INFRASTRUKTURA JAVNIH PUNJAČA

Infrastrukturni element, koji služi za napajanje javnih punjača električnih automobila predstavlja stanica za punjenje električnih automobila ili punionica.

Na Grbavici, javni punjači (punionice) mogu se postaviti na različita mesta, na primer u okviru stambenog prostora (ukoliko ima parkinga), na taksi stajalištima ili javnim parkiralištima. Takođe postoji mogućnost instaliranja u okviru benzinskih stanica. Pored raspoloživog parking prostora, kriterijum za njihovo postavljanje je i da koriste postojeću električnu mrežu za svoje napajanje ili obnovljive izvore. Jedan od izazova za takvu infrastrukturu je nivo potražnje, vreme punjenja i način obračuna isporučene energije. Kod planiranja postavljanja javnih punjača na teritoriji Grbavice, mora

se napraviti strategija po kojoj će biti izvršena podela zona, zatim optimalan broj punjača, kao i opravdanost postavljanja na određenoj lokaciji.

Za javne punjače izabrani su modeli kompanije ABB i to isključivo polubrzi i brzi punjači [4]. Korišćeni model za brzo punjenje je tipa AC, snaga 43 kW, *Terra 53 CJG*, dok je model za polubrzno punjenje tipa AC, snaga 22 kW, *Terra 23 CT*.

Kod postavke punionica mora se voditi računa i o ukupnoj instalisanoj snazi, koja mora da odgovara kapacitetima distributivne mreže. Kako bi se opterećenje mreže usled novih priključenih potrošača (javnih punjača) optimalno raspodelilo, sve zone će biti povezane na posebne transformatorske stanice (ukoliko je to moguće), a instalisana snaga nijedne zone ne sme preći 130 kW (3 brza punjača). Nakon što je izvršen plan i raspored zona, izvršeno je pozicioniranje punjača i određena je napajacka transformatorska stanica za svaku zonu. Planirano je ukupno 7 lokacija za postavljanje punjača (L1-L7), a njihov pregled dat je u tabeli 1. Predviđeno je postavljanje 6 brzih i 7 polubrzi punjača, ukupne instalisane snage od 412 kW. U nastavku su dati detalji svake lokacije:

L1-Ulica Bulevar Oslobođenja 98 sa dvorišne strane. Dužina kabela je 80 m i TS se nalazi u dvorištu.

L2-Ulica Vladimira Nikolića 12. Dužina kabela je 53 m, priključuje se na istu TS kao i L1.

L3-Ulica Milana Simovića 8. Dužina kabela je 33 m a između uređaja 5 m. TS je u dvorištu iza zgrade.

L4-Ulica Miše Dimitrijevića 30. Dužina kabela je 190 m, priključuje se na TS u ulici Miše Dimitrijevića 9A, kao i FN elektrana.

L5-Ulica Braće Ribnikar 4. Dužina kabela je 20 m, a između uređaja 5 m. TS je u dvorištu zgrade.

L6-Ulica Bulevar oslobođenja 76. Dužina kabela je 80 m, a između uređaja je 5 m. TS je sa dvorišne strane.

L7-Ulica Alekse Šantića 57. Dužina kabela je 85 m, a između uređaja 5 m. TS je u ulici Alekse Šantića 26, gde je priključena FN elektrana.

Tabela1. Pregled svih lokacija i njihovih snaga

Položaj	Brzo punjenje	Polubrzno punjenje	Snaga [kW]
L1	1	-	43
L2	1	-	43
L3	1	1	65
L4	-	1	22
L5	1	1	65
L6	1	2	87
L7	1	2	87
Ukupno	6	7	412

Kako se sve stanice napajaju iz različitih TS, problem preopterećenja je izbegnut. Povezivanje stanice za punjenje i TS vrši se kablovskim vodovima ukopanim u zemlju, tj. D razvodom i to primenom TN-C sistema napajanja, četvorožičnim napajanjem, tri fazna provodnika i četvrti koji ujedno predstavlja i neutralni i zaštitni provodnik.

Na slici 4 dat je prikaz TS na koje je predviđeno priključenje javnih električnih punjača na teritoriji Grbavice (uokvirene belim). Treba napomenuti da ovaj

segment nije u nadležnosti projektanta, već u nadležnosti lokalnog elektro-distributivnog preduzeća, koje propisuje uslove priključenja, dimenzioniše kablove, zaštitne uređaje i ormar mernog mesta koji služe za povezivanje stanice. Kablovska i zaštitna oprema u samoj stanici za punjenje bira se na osnovu proračuna, koji je opisan u narednom poglavlju [5].



Slika 4. Prikaz TS za električne punjače na teritoriji Grbavice

6. POVEZIVANJE JAVNIH PUNJAČA

6.1. Proračun elemenata

Svi kablovi koji se koriste u stanici za punjenje određuju se na osnovu pravila propisanih standardom IEC 60364-5-523. Predviđeno je da su svi kablovi položeni u zemlju, a da je izolacija svakog izrađena od umreženog polietilena. Presek provodnika kabla, koji će biti korišćen mora biti u skladu sa vrednostima standarda IEC 60364-5-523, a bira se na osnovu vrednosti dobijenih proračunom. Sve proračunate vrednosti su date u tabeli 2. Radi ekonomičnosti, može se uraditi standardizacija kablova, tako da se na svim lokacijama koristiti kabl XPOO preseka 70 mm², položen u zemlju. Ovakva standardizacija urađena je na stranu sigurnosti, s obzirom da je uzet najveći presek kabla.

Tabela 2. Brojne vrednosti pri proračunu elemenata (kablova i zaštitnih uređaja)

Zona	I ₁ (A)	I ₂ (A)	I ₃ (A)	I _{TD} (A)	I _{TTD} (A)	S (mm ²)	I _{ZAST} (A)
L1	63	-	-	63	101	25	80
L2	63	32	-	95	122	35	100
L3	63	32	-	95	122	35	100
L4	-	32	-	32	45	6	35
L5	63	32	-	95	122	35	100
L6	63	32	32	127	178	70	160
L7	63	32	32	127	178	70	160

6.2. Provera preseka provodnika na pad napona

U tabeli 3 dati su ključni parametri predviđenog kabla i izvršen proračun ukupnog pada napona. Kao što se može videti, nema prekoračenja dozvoljenog pada napona, odnosno sve vrednosti su u granici od 1%, tako da kablovi zadovoljavaju uslove potrošača, pad napona je zanemarljiv, pa ujedno predstavljaju i konačan odabir za povezivanje punionica sa TS na Grbavici.

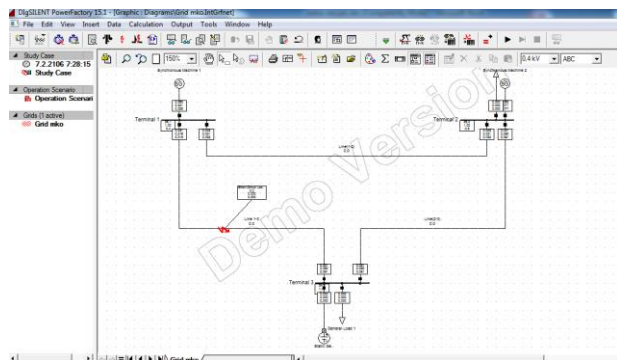
Tabela 3. Brojne vrednosti prilikom proračuna pada napona (ustanovljen poprečni pesek od 70 mm²)

Zo na	I (A)	S (mm ²)	r (Ω/km)	x (Ω/km)	l (m)	cos φ	tan φ	Δu (%)
L1	63	70	0.15	0.08	80	0.95	0.33	0.36
L2	95	70	0.15	0.08	53	0.95	0.33	0.36
L3	95	70	0.15	0.08	33	0.95	0.33	0.23
L4	32	70	0.15	0.08	190	0.95	0.33	0.44
L5	95	70	0.15	0.08	20	0.95	0.33	0.14
L6	127	70	0.15	0.08	80	0.95	0.33	0.74
L7	127	70	0.15	0.08	85	0.95	0.33	0.79

7. DigSilent SIMULACIJA KVARA

Za ovako postavljenu strukturu javnih punionica, a s obzirom na način proračuna i zadovoljene zadate kriterijume, može se zaključiti da će raditi sigurno i pouzdano. Međutim, u distributivnoj mreži može doći do pojava kvarova, bilo kratkotrajnih, bilo onih koji izazivaju kompletan ispad napajanja. U ovom radu će biti razmatrani samo oni, koji su prolaznog karaktera i koji se manifestuju pojavom propada napona na mestu priključenja punionica na mrežu. U tim slučajevima, dolazi do reagovanja zaštite i proces punjenja baterija se trajno prekida, bez obzira što je propad napona izuzetno kratko trajao (200 ms – 3 s). To predstavlja problem, pa ga je potrebno dodatno istražiti.

Za principijelno istraživanje postavljenog problema iskoristiće se softverski alat *DigSilent*, u kom je samostalno napravljen jedan hipotetički segment distributivne mreže [6]. Ovaj deo ima tri sabirnice, tri voda, na koje je povezana jedna baterija koja se puni u punionici, dva sinhrona generatora i proizvoljna dva opterećenja, što je predstavljeno na slici 5. Sistem je dimenzionisan za srednji napon od 35 kV. Simuliran je kratak spoj na vodu 1-3 i posmatran je njegov uticaj na bateriju (punjač) i sistem.



Slika 5. Prikaz sistema u DigSilent-u sa prikazanim mestom simuliranog kratkog spoja.

Na slici 6 prikazan je dijagram sa parametrima dvopolnog kratkog spoja sa zemljom. Pre pojave kvara, baterija (električnog vozila) punila se strujom od 0,117 kA. Prilikom pojave kvara (početak simulacije) struja je pala na nulu, što je očekivano zbog prorade zaštite koja se aktivirala u momentu prepoznavanja kvara. Vidi se da subtranzijentna snaga S_k za faze a,b,c iznosi u [MVA]: 0; 10,55; 11,36, respektivno, dok je struja kratkog spoja u subtranzijentnom periodu I_k po fazama u [kA°]: 0/0°; 0,52/177,75°; 0,56/42,44°, respektivno.

Može se zaključiti da pojava kvara u mreži može uticati na rad punionica, čak i ako se radi o veoma kratkim,

prolaznim kvarovima. Simulacije u softverskom alatu *DigSilent* pokazuju da se pojavljuju struje kvara, koje bi izazvale reagovanje zaštite ovih uređaja, odnosno došlo bi to njihovog ispada, prestanka procesa punjenja.



Slika 6. Dijagram struje kratkog spoja

8. ZAKLJUČAK

Važan preduslov za širu upotrebu električnih vozila predstavlja postavljanje široke infrastrukture punionica za baterije. Ako se one napajaju iz krovni FN elektrana, na području Novog Sada u delu Grbavica, moguće je smanjiti potrošnju za oko 26% i za tu vrednost umanjiti emisiju CO₂ u TE.

9. LITERATURA

- [1] M.Steen, „Greenhouse Gas Emissions From Fossil Fuel Fired Power generation Systems“, EUR 19754 EN, Joint Research Centre, Inst. for Advanced Materials, EC, 2000.
- [2] <https://pushevs.com/2016/11/23/electric-cars-range-efficiency-comparison/>
- [3] J.Janković, „Urbane solarne elektrane-studija slučaja za Grbavicu u Novom Sadu“, Diplomski rad, Fakultet tehničkih nauka, Novi Sad, 2018.
- [4] <https://www.elektropunjac.com/kategorije-proizvoda/javni-punjaci/>
- [5] D. Head, P. Kind, „Electric Vehicle Charging Station Program and Installation Guidelines“, Sonoma so, Sacramento, 2011.
- [6] <https://www.digsilent.de/en/>

Kratka biografija:



Jovana N. Janković, rođena je 20.02.1994. god. u Novom Sadu. Diplomirala je na Fakultetu tehničkih nauka u Novom Sadu 2018.godine. Upisala master studije 2018/2019 god iz oblasti Elektrotehnika i računarstvo - Distribuirani elektroenergetski resursi, a master rad je odbranila jula 2019. god.



Vladimir A. Katić, rođen je 1954. god. u Novom Sadu. Diplomirao je na Fakultetu tehničkih nauka u Novom Sadu 1978. god., a magistrirao i doktorirao na Elektrotehničkom fakultetu Univerziteta u Beogradu 1981. i 1991. god., respektivno. Oblasti interesovanja su mu energetska elektronika, električna vozila, obnovljivi izvori energije i kvalitet električne energije.

MODEL STATIČKOG VAR KOMPENZATORA PRI REGULACIJI NAPONA**STATIC VAR COMPENSATOR MODEL FOR VOLTAGE CONTROL**Mihajlo Govorčin, Marko Vekić, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj- Ovaj rad analizira neke od osnovnih FACTS uređaja sa naglaskom na SVC-u. Opisane su prednosti i uloge FACTS uređaja. Dati su rezultati simulacije dinamičkog odziva SVC-a pri regulaciji napona u sabirnici priključenja.

Ključne reči: FACTS uređaji, kompenzacija reaktivne snage, nadzor napona, SVC

Abstract- This paper deals with FACTS devices and their applications. The advantages of FACTS devices with emphasis on SVC, are described. The results of dynamic SVC simulation are presented.

Keywords: FACTS devices, reactive power compensation, voltage control, SVC

1. UVOD

Tehnologija FACTS uređaja (Flexible AC Transmission Systems) otvara nove mogućnosti [1] za upravljanje tokom snage i povećanje iskoristivosti postojećih, kao i novih i nadograđenih prenosnih vodova. Mogućnost nadzora struje, a time i snage, u prenosnom vodu omogućava povećanja kapaciteta postojećih vodova. Te mogućnosti proizlaze iz sposobnosti FACTS uređaja da nadziru međusobno povezane parametre koji regulišu rad prenosnih sistema, uključujući rednu impedansu, otočnu impedansu, struju, napon, fazni stav i prigušenje oscilacija.

U FACTS uređaje spadaju: otočni Var kompenzator (SVC), tiristorski upravljana prigušnica (TCR), tiristorski prekidana prigušnica (TSR), tiristorski uključivan kompenzator (TSC), statički sinhroni kompenzator (STATCOM), tiristorski uključivan redni kompenzator (TSSC), statički redni sinhroni kompenzator (SSSC) itd. U ovom radu će naglasak biti stavljen na otočno povezane FACTS uređaje, u prvom redu SVC.

2. Otočni statički kompenzator

Kompenzacija se koristi da bi se uticalo na prirodne električne karakteristike voda, da bi se povećala nazivna prenosna snaga i da bi se nadzirao profil napona duž voda. Najpoznatiji otočni kompenzatori su SVC i STATCOM.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Marko Vekić, docent.

Statički Var kompenzator (SVC) je može da deluje kao izvor ili ponor reaktivne snage [2] koji je podešen za razmenu kapacitivne ili induktivne struje kako bi se održali ili nadzirali specifični parametri elektroenergetskog sistema. Dejstvo SVC-a je zasnovani na induktivnim i kapacitivnim elementima u kombinaciji sa prekidačima energetske elektronike. Na taj način odziv i dinamika SVC-a je značajno brža od sinhronih kompenzatora koji su u suštini sinhrona mašine. U SVC spada nekoliko klasa FACTS uređaja kao što su: TCR, TSR i TSC.

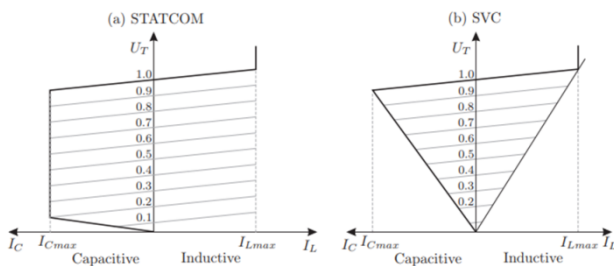
STATCOM radi kao otočno spojeni statički var kompenzator čija se kapacitivna ili induktivna struja može kontrolisati nezavisno od napona. STATCOM je kontrolisani izvor reaktivne snage. On obezbeđuje naponsku podršku generisanjem ili prigušenjem reaktivne snage u sabirnici priključenja bez potrebe za velikim spoljnim prigušnicama ili kondenzatorskim baterijama.

3. Poređenje otočnih kompenzatora

SVC i STATCOM su slični po svojim sposobnostima kompenzacije, ali osnovna načela rada su različita. STATCOM deluje kao otočno povezani izvor napona, dok SVC deluje kao upravljiva reaktivna admitansa koja je otočno povezana. Ova razlika se ogleda u funkcionalnim karakteristikama STATCOM-a. STATCOM, kao nezavisan naponski izvor ima veću moć kompenzacije i veći opseg primena od onih koje se mogu postići sa SVC-om, ali SVC je jednostavniji i jeftiniji za primenu.

U linearnom radnom opsegu U-I karakteristike (Slika 1) i sposobnosti kompenzacije STATCOM-a i SVC-a su slične [3]. Što se tiče nelinearnog radnog opsega, STATCOM može da kontroliše svoju izlaznu struju nezavisno od napona mreže, dok maksimalna kompenzujuća struja SVC-a linalno opada sa AC naponom, pošto je SVC u suštini kontrolisana susceptansa. Tako je STATCOM delotvorniji od SVC-a u obezbeđivanju naponske podrške pri velikim sistemskim smetnjama tokom kojih bi naponska odstupanja bila daleko izvan linearnog radnog opsega kompenzatora. Sposobnost STATCOM-a da održava punu kapacitivnu izlaznu struju pri niskim naponima sistema čini ga efikasnijim od SVC-a u poboljšanju stabilnosti.

Vreme odziva i propusni opseg regulacione petlje napona STATCOM-a su takođe značajno bolji od onih kod SVC-a. U situacijama kada je neophodno obezbediti dotok aktivne snage, STATCOM je u mogućnosti da poveže odgovarajuće skladište energije (veliki kompenzator, baterija) odakle može povući aktivnu snagu i isporučiti ju putem pretvarača kao AC snagu sistemu, dok SVC nema tu sposobnost.



Slika 1: U-I karakteristike STATCOM-a (a) i SVC-a (b)

Glavna prednost SVC-a je u kompenzaciji na visokom naponu gde bi pogon STATCOM-a bio znatno skuplji, kao i načelno jednostavnija i jeftinija izvedba SVC-a koji je sposoban vrlo uspešno vršiti kompenzaciju sve do značajnog pada napona.

4. Statički redni kompenzatori

Promenljiva redna kompenzacija je veoma delotvorna kako u nadzoru protoka snage tako i u poboljšanju stabilnosti. Sa rednom kompenzacijom, ukupna impedansa voda može se proizvoljno smanjiti, što utiče na tok snage ($P = U^2/X \cdot \sin \delta$). Ova sposobnost kontrole protoka snage može se efikasno koristiti za povećanje granice stabilnosti i za prigušenje oscilacija snage. Neke od vrsta statičkih rednih kompenzatora su: Tiristorski uključivan redni kompenzator (TSSC), Tiristorski upravljana redna kapacitivnost (TCSC), GTO Tiristorski upravljani redni kondenzator (GCSC) i Statički redni sinhroni kompenzator (SSSC).

5. Način i mesto postavljanja SVC-a

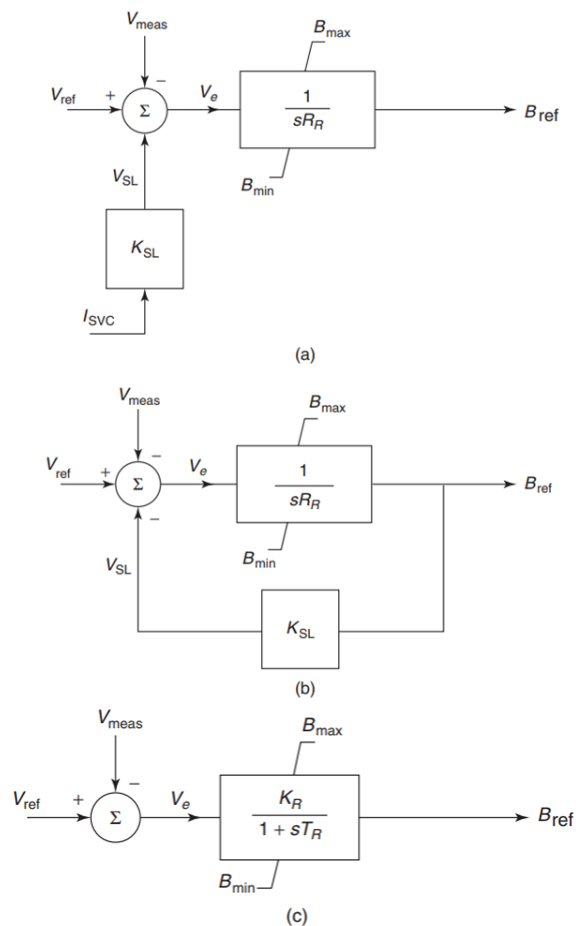
Kompenzator može da se postavi na različitim mestima voda. Postavljanjem SVC-a na sredini voda i time razdvajanje voda na dva dela, napon u ovoj tački može se kontrolisati tako da ima istu vrednost kao i naponi na kraju voda. Time se povećava maksimalni prenos snage. Ako je kompenzator postavljen na kraju voda odočno sa opterećenjem, moguće je regulisati napon na tom kraju i time sprečiti nestabilnost napona uzrokovanu promenama opterećenja. Pošto je otočna kompenzacija u stanju da moduliše tok struje, može se povećati granica prelazne stabilnosti i obezbediti prigušenje oscilacija snage (tzv. aktivno prigušenje).

6. Regulacija napona pomoću SVC-a

Regulator napona SVC obrađuje izmerene veličine merenog sistema i proizvodi izlazni signal koji je srazmeran željenoj kompenzaciji reaktivne snage. Izmerene upravljačke promenljive porede se sa referentnim signalom, obično V_{ref} , a signal greške je ulaz funkcije prenosa regulatora. Izlaz regulatora je željena vrednost susceptanse B_{ref} kojom se postiže zadata referentna vrednost napona u sabirnici priključenja. Obično se ugrađuje mali nagib [4] ili pad (3-5%) u karakteristiku SVC-a kako bi se postigle određene prednosti. Moguće primene ovog nagiba u modelu regulatora napona prikazane su na slici 2; na slici 2a prikazan je slučaj sa povratnom spregom po struji SVC-a. Izmerena vrednost struje je pomnožena sa faktorom K_{SL} koji predstavlja nagib (droop) stuje pre nego što se sabere sa naponom V_{SL} . V_{SL} je takav da odgovara povećanju referentnog napona za induktivne SVC struje i smanjenje

referentnog napona za kapacitivne SVC struje. Jednostavno integralno dejstvo se najčešće koristi u naponskim regulatorima. Primetimo da u kombinaciji sa padom (droop-om) čini zapravo PI regulator. Brzina odziva η_R ukazuje na vreme koje je potrebno SVC-u da se kreće kroz čitav opseg reaktivne snage, tj. od potpuno kapacitivnog do potpuno induktivnog stanja kao u cilju regulacije napona.

U određenim slučajevima, može biti teško da se dobije veran strujni signal. Ovo se dešava kada SVC radi blizu svog nestacionarnog stanja, tj. pri nultoj reaktivnoj snazi. Strujni signal tada sadrži harmonijske komponente i osnovnu komponentu koja odgovara stvarnim gubicima u SVC-u. Da bi se prevazišao ovaj problem, u nekim SVC regulatorima se koristi procenjena reaktivna snaga u regulaciji umesto signala struje. Signal reaktivne snage izračunava se množenjem faznih struja SVC-a sa osnovnim harmonikom napona koji zaostaje za stvarnim naponom za 90° .



Slika 2: Moguće primene strujnog nagiba (droop-a) u regulatoru napona: a) integrator sa povratnom spregom struje; b) integrator sa povratnom spregom pri padu susceptanse; c) pojačanje vremenske konstante

Druga mogućnost koja se može lako postići je povratna sprega sa droop-om po susceptansi koja je prikazana na slici 2b. Podrazumeva se da napon SVC sabirnice ostaje blizu 1 r.j.p a je struja SVC-a strogo jednaka B_{ref} . V_{SVC} se jednostavno može izraziti kao B_{ref} u r.j. Upravljački sistem zatvorene petlje na slici 2b može se pojednostaviti kao na slici 2c. Pojačanje K_R se naziva statičko pojačanje, koje je definisano kao inverzna vrednost droop-a struje.

$$K_R = \frac{1}{K_{SL}}$$

Takođe

$$T_R = \frac{\eta_R}{K_{SL}}$$

Pojačanje K_T , predstavlja dinamičku prirodu regulatora napona i definiše se kao:

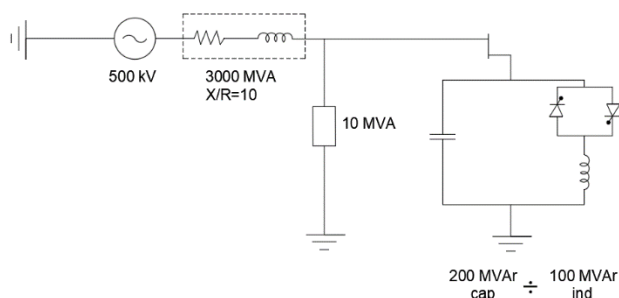
$$K_T = \frac{K_R}{T_R}$$

Droop struje regulatora napona osigurava linearnost između napona SVC-a i struje SVC-a u upravljačkom opsegu. Povratna sprega po padu susceptanse obezbeđuje linearnost između susceptanse SVC-a i napona koja se odražava u neznatno nelinearnom odnosu napona i struje u slučaju varijacije napona. Međutim, neusaglašenost između ove dve predstave nije značajna, jer je droop obično veoma mali, a varijacije napona SVC-a generalno nisu velike. Dinamičko delovanje SVC-a može biti predstavljeno od strane bilo kog od ova dva pristupa.

Prednost modela droop-a struje je da karakteristika ustaljenog stanja (opisana uz pomoć K_R) i dinamička karakteristika (uvažena preko R_p) mogu biti nezavisno određene. U modelu pojačanja vremenske konstante, karakteristike prelaznog i ustaljenog stanja su međusobno povezane kroz parametra K_R i T_R . Model pojačanja vremenske konstante je naširoko korišćen u SVC modelovanju za SVC ispitivanja, iako integrator sa padom struje predstavlja fizičku realizaciju većine postavljenih SVC-ova.

7. Simulacija dinamičkog odziva SVC-a

Simuliraćemo sistem sa slike 3 predstavljen pofaznom šemom. Pretpostavljamo da je dati sistem uravnotežen.



Slika 3: Simulirani sistem predstavljen pofaznom šemom
Usvojili smo bazne vrednosti:

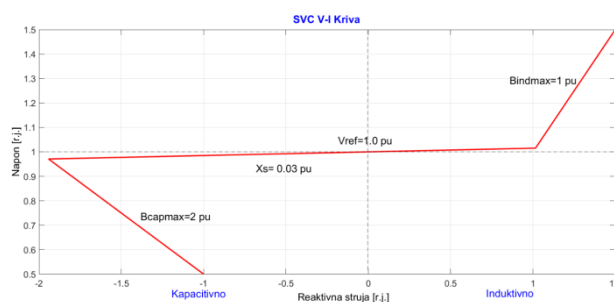
$$S^b = 100\text{MVA} \quad , \quad V^b = 500\text{kV}$$

Pošto je:

$$S^b = \sqrt{3} \cdot V^b \cdot I^b = \sqrt{3} \cdot V^b \cdot B^b \cdot V^b$$

$$\Rightarrow B^b = \frac{S^b}{\sqrt{3}(V^b)^2}$$

Za vrednost kondenzatorske baterije SVC-a usvojeno je $B_{cap}^{max} = 2 \cdot B^b$, dok je za prigušnicu SVC-a usvojeno $B_{ind}^{max} = 1 \cdot B^b$. Za nagib karakteristike SVC-a je usvojen pad (droop) od 0.03. Karakteristika SVC-a je prikazana na slici 4.



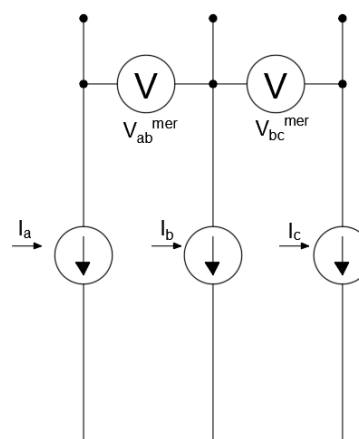
Slika 4: Karakteristika SVC-a

Nagib X_S jasno ukazuje da će postojati izvesno malo odstupanje stvarne vrednosti napona koja se postiže dejstvom SVC-a u odnosu na željenu.

Zbog ovog malog odstupanja u naponu postiže se značajno smanjenje neophodnih struja, odnosno reaktivne snage SVC-a da bi se postigao napon blizak željenom.

Model SVC-a koji smo primenili je pojednostavljen u smislu da je dejstvo SVC-a modelovano upravljanim strujnim izvorima (slika 5).

Odgovarajuće fazne struje su dobijene na osnovu regulisane promenljive susceptanse (koja se dobije kao izlaz regulatora napona) i odgovarajućih faznih napona mreže.

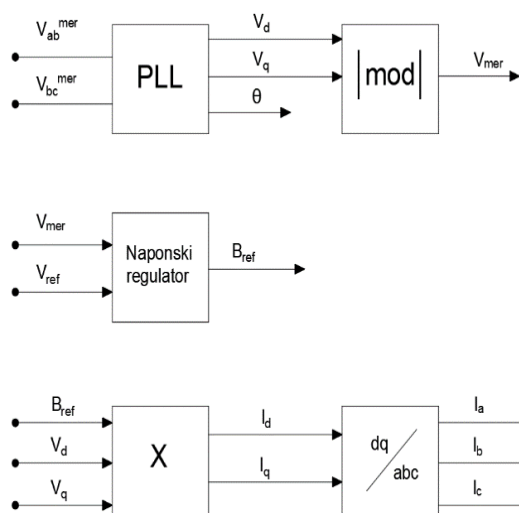


Slika 5: Pojednostavljen model SVC-a prikazan preko nadziranih strujnih izvora

Na osnovu merenih napona se iz bloka PLL-a [5] dobije modul fazora napona i njegov ugao (slika 6).

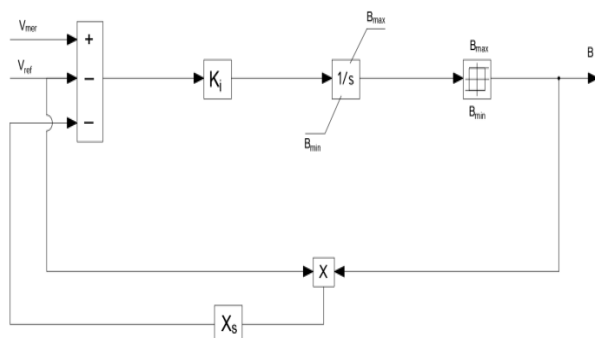
Naponski regulator daje neophodnu vrednost susceptanse da bi se tako dobijeni izmereni fazor regulisao na referentnu vrednost.

Množenjem susceptanse sa komponentama napona po d i q osi dobijemo odgovarajuće komponente struja, pa inverznom Park-ovom transformacijom dobijemo fazne struje i time je model zatvoren.



Slika 6: Postupak dobijanja faznih struja

Sklop naponskog regulatora dat je na slici 7.



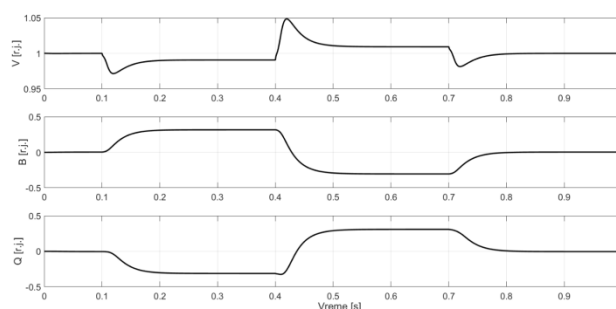
Slika 7: Šema naponskog regulatora

Integralno pojačanje dato je kao $K_i = \frac{1}{\eta_R}$, pri čemu je η_R definisano kao vreme odziva SVC-a s jednog kraja radnog opsega (kapacitivnog) do drugog kraja (induktivnog) i obično iznosi nekoliko ms. Usvojimo da je $\eta_R = 3\text{ms}$, pa je $K_i = 333$, ali ćemo usvojiti $K_i = 300$. X_s kao što smo već napomenuli iznosi 0.03 [r.j.]. Za referentnu vrednost napona V^{ref} usvojena je vrednost, $V^{\text{ref}} = 1$ [r.j.].

Posmatraćemo odziv SVC-a u slučaju kada napon mreže u trenutku 0.1s padne sa 1 [r.j.] na 0.97 , zatim u trenutku 0.4s poraste na 1.03 [r.j.] i na kraju u trenutku 0.7s se vrati na 1 [r.j.]. Prikazaćemo odziv napona SVC-a, njegove susceptanse i reaktivne snage.

Sa slike 9 možemo videti da napon SVC-a ne dostiže referentnu vrednost 1 [r.j.] nakon što mrežni napon padne na 0.97 [r.j.] već se ustali na 0.99 [r.j.].

Ovo je posledica pomenutog nagiba SVC karakteristike X_s . Isto važi i za preskoke napona u induktivnoj oblasti.



Slika 9: Odzivi napona SVC-a, njegove susceptanse i reaktivne snage

8. Zaključak

Ovaj rad je dao pregled uloga i primena FACTS uređaja sa naglaskom na statički VAR kompenzator (SVC). Ukazane su prednosti i mane SVC-a u odnosu na još jedan otočno povezan kompenzator - STATCOM. Na osnovu rezultata simulacije, jasno je da SVC može uspešno vršiti regulaciju napona u sabirnici priključenja sa zadovoljavajućom dinamikom. Referentna vrednost napona se ipak ne dostiže u potpunosti zbog namerno umetnute droop karakteristike koja omogućava postizanje gotovo referentnih vrednosti napona, ali sa mnogo manjim utroškom reaktivne snage, odnosno manjim reaktivnim strujama.

9. Literatura

- [1] E. Acha, V.G. Agelidis, "Power Electronic Control in Electrical Systems", Newnes 2002, ISBN 0750651261
- [2] Oscar Skoglund, "Dynamic voltage regulation using SVCs", Stockholm, 2013
- [3] Gabriela Glanzmann, "FACTS Flexible Alternating Current Transmission Systems
- [4] R. Mohan Mathur, Rajiv K. Varma, "Thyristor-based FACTS controllers for electrical transmission systems, 2002, ISBN 0471206431
- [5] M. Vekic et al., "Multi-Resonant Observer PLL with Real-Time Estimation of Grid Unbalances", International Journal of Electrical Power and Energy Systems, Elsevier June 2019, ISSN 01420615

Kratka biografija:



Mihajlo Govorčin rođen je u Novom Sadu 1995. godine. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine odbranio je 2018.god.



Marko Vekić je docent na Fakultetu tehničkih nauka u Novom Sadu, na Katedri za Energetsku elektroniku i pretvarače. Oblast interesovanja su energetska elektronika u prenosnim i distributivnim mrežama i kvalitet električne energije.

GENERISANJE KODA ZA VIZUALIZACIJU REZULTATA ELEKTROENERGETSKIH PRORAČUNA UZ UPOTREBU OBLASNO SPECIFIČNOG JEZIKA**CODE GENERATION FOR VISUALIZING THE RESULTS OF THE ELECTRIC POWER CALCULATIONS USING A DOMAIN SPECIFIC LANGUAGE**

Vladislav Simić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu je predstavljena analiza mogućnosti stvaranja jezika kojim se modeluje izvještaj za rezultate proračuna elektroenergetskih funkcija kao i generisanje izvornog koda na osnovu modela tog izvještaja. Analizom softvera za menadžment elektroenergetskih sistema usvojeni su koncepti na kojima se zasniva domenski specifičan jezik za modelovanje izvještaja rezultata proračuna. Rješenje opisano u radu implementirano je uz oslonac na Microsoft DSL tehnologiju. Cilj ponuđenog rješenja je da olakša i ubrza razvoj segmenata kompleksnog elektroenergetskog softvera, odnosno razvoj izvještaja za rezultate proračuna.

Ključne reči: Modelovanje, metamodel, domenski specifičan jezik, generator koda, UML, DSL

Abstract – The paper presents an analysis of the possibility of creating a language which models the report for the results of the calculation of electrical energy functions as well as generating the source code based on the model of that report. An analysis of the software for the management of electric power systems has provided concepts on which the domain specific language for the modeling of the report on results of the calculations is based. The solution described in the paper is implemented with the support of Microsoft DSL technology. The goal of the offered solution is to facilitate and accelerate the development of segments of complex power software, that is, the development of the report on the results of the calculations.

Keywords: Modeling, metamodel, domain specific language, code generator, UML, DSL

1. UVOD

Jedan od osnovnih zadataka prilikom implementacije softvera predstavlja odabir programskog jezika. Na njegov izbor utiču mnogobrojni faktori kao što su poznavanje samih tehnologija, kompanijska pravila i procedure, kao i implementacioni zahtjevi koji se očekuju od rješenja. Programski jezici predstavljaju osnovni alat za razvoj softvera i dijele se na jezike opšte namjene i jezike specifične za domen [1].

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Nemanja Nedić, docent.

Druge vrsta izazova koja se javlja prilikom razvoja softvera predstavlja proces modelovanja rješenja domenskog problema. U većini slučajeva, koncepti jezika opšte namjene nisu usklađeni sa konceptima domena problema. To ima za posledicu složenije modelovanje rješenja problema kao i samu implementaciju rješenja. Sa druge strane, korištenjem jezika specifičnog za domen omogućeno je trivijalno modelovanje rješenja, obzirom da je jezik zasnovan na konceptima domena za koji se projektuje [2].

Tema ovog rada jeste analiza vezana za mogućnost implementiranja domenski specifičnog jezika kojim je moguće modelovati u tekstualnoj ili grafičkoj formi izvještaje za rezultate proračuna elektroenergetskih funkcija [3]. Sa druge strane rad se bavi i analizom mogućnosti implementiranja generatora koda koji koristi koncepte prethodno pomenutog jezika.

2. POSTAVKA PROBLEMA

Razlog za bavljenje ovim problemom proizilazi iz potrebe da se implementira univerzalno rješenje pomoću koga bi domenski eksperti na relativno trivijalan način projektovali izvještaje rezultata elektroenergetskih proračuna. Rezultati koji se prikazuju krajnjem korisniku organizovani su u sličnim formama posmatrajući različite funkcije. To znači da prikaz rezultata svake funkcije projektujemo koristeći iste elemente grafičkog korisničkog interfejsa.

Analogno načinu prikazivanja rezultata i izvorni kod ima identičnu strukturu za svaki od tipova elementa elektroenergetskog sistema. Izvorni kod kojim je implementiran softver je uglavnom repetitivan i generičan pri posmatranju određenih segmenata.

Na osnovu toga, uvođenjem domenski specifičnog jezika olakšalo bi se modelovanje izvještaja, a uvođenjem generatora koda koji se oslanja na koncepte tog jezika omogućila bi se automatska transformacija modela na izvorni kod i izbjeglo bi se manuelno kodiranje segmenata koda koji su repetitivni i generični. Bitno je napomenuti da kada se kaže da je izvorni kod repetitivan ne misli se da na više mjesta postoji identičan kod, već da su različiti elektroenergetski elementi opisani različitim klasama koje imaju istu strukturu koda, dok njihova svojstva definišu elementi za koje su klase kreirane.

Neke od osnovnih funkcija za elektroenergetske proračune su: proračuni tokova snaga, estimacija stanja, proračuni optimalnih tokova snaga, analiza optimalne topološke promjene, analiza sigurnosti prilikom ispada i druge [3]. Sa druge strane, osnovne elemente elektroenergetskog

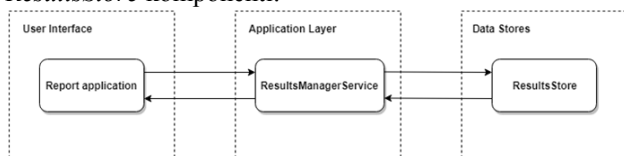
sistema predstavljaju generatori, vodovi, transformatori, kondenzatori, potrošači [4]. Primjera radi, posmatrajući generator, parametri koji se proračunavaju nad njim su aktivna, reaktivna i prividna snaga [4]. Rezultati proračuna ovih parametara u najjednostavnijem obliku korisniku se prikazuju tako što se rezultati za svaki tip elementa organizuju u kartice (eng. *Tab*), a u okviru svake kartice nalazi se tabela, gdje kolone tabele predstavljaju parametre elemenata. Takav jedan primjer prikazivanja rezultata predstavljen je na slici 2.1.

Nodes	Branches	Shunts	Switching devices						
Generators	Interconnections	Loads	Capacitors	Compensators	Energy Storages				
Filter									
Element	Type	Loading (%)	S (MVA)	P (MW)	Q (MVar)	I (A)	Power Factor	Losses Sensitivity Factor	Quality
Transmission East									
Arta									
Arta									
Arta									
Arta									
Arta									
Arta Major									
Corona									
GEN_14439670311668046	PV	9.85	5.5	5.3	-1.4	287.81	0.97 lead	1.0000	Good
GEN_14439670311668049	PV	10.60	6.5	5.3	-1.7	351.11	0.82 lead	1.0000	Good
GEN_14439670311668051	PV	61.62	37.7	37.7	6.8	2036.44	1.00 lag	1.0217	Good
GEN_14439670311668052	PV	61.62	37.7	37.7	6.8	2036.70	1.00 lag	1.0216	Good
GEN_14439670311668047	PV	20.81	5.1	5.1	0.1	459.04	1.00 lag	1.0199	Good
GEN_14439670311668048	PV	42.59	5.1	5.1	-0.1	459.03	1.00 lead	1.0199	Good
ApplKwh_EMS_Tot_3	PQ	29.15	7.9	7.7	-1.7	680.06	0.98 lead	1.0198	Good
Generators									
Champion									
GEN_14439670311668053	PV	7.76	11.4	11.4	-1.3	432.84	0.99 lead	1.0187	Good
Corona Kentucky									
GEN_14439670311668045	PV	6.88	2.3	2.2	0.7	122.18	0.95 lag	0.9808	Good
Corona Australia									
Crow									
Crown									
Crown									
GEN_14439670311668037	PQ	53.96	6.5	6.5	0.0	590.05	1.00 lead	0.9713	Good
GEN_14439670311668039	PV	16.91	4.5	4.5	1.0	586.88	0.99 lag	0.9712	Good
GEN_14439670311668039	PV	13.28	9.7	9.7	-0.4	534.83	1.00 lead	0.9613	Good
GEN_14439670311668040	PV	15.93	9.7	9.7	-0.4	534.73	1.00 lead	0.9612	Good
GEN_14439670311668041	PV	34.32	10.3	10.3	-0.4	907.31	0.97 lead	0.9609	Good
GEN_14439670311668042	PV	15.77	9.8	9.8	-0.3	525.42	1.00 lead	0.9612	Good
GEN_14439670311668043	PV	16.07	9.8	9.7	-1.4	518.12	0.99 lag	0.9613	Good

Slika 2.1 Prikaz rezultata proračuna

3. ANALIZA DOMENA

Softver koji omogućava izvršavanje elektroenergetskih proračuna i za koji se projektuje ovaj generator koda je implementiran u troslojnoj klijent-server arhitekturi (slika 3.1). U prezentacionom sloju nalazi se *Report* aplikacija koja prikazuje rezultate proračuna. U srednjem ili aplikativnom sloju nalazi se *ResultsManagerService*. Ova komponenta na zahtjev prezentacionog sloja dobavlja rezultate iz skladišta i u određenoj strukturi ih šalje *Report* aplikaciji. Skladištenje podataka povjereno je *ResultsStore* komponenti.



Slika 3.1 Arhitektura softvera

Rezultati proračuna za svaki elektroenergetski element modelovani su zasebnom klasom. Vrijednosti atributa u okviru ovih klasa predstavljaju rezultate proračuna i čuvaju se u okviru *ResultsStore* komponente. Pristup tim vrijednostima omogućen je upotrebom *Generic Data Access (GDA)* interfejsa koji *ResultsStore* komponenta izlaže drugim komponentama. *GDA* interfejs omogućava pristup podacima bez ikakvog poznavanja logičke šeme baze podataka [5]. Da bi *GDA* „znao“ da pristupi određenoj klasi ili njenom atributom, potrebno je na određeni način svaku klasu ili njen atribut jednoznačno odrediti. To se postiže uvođenjem jedinstvenog identifikatora, tzv. *ModelIdentificator*-a koji predstavlja 64-bitnu vrijednost. Mapiranjem jedinstvene 64-bitne vrijednost na određenu klasu ili atribut klase dobijaju se metapodaci koji opisuju konkretne podatke. Fomiranjem *ModelIdentificator*-a prema predefinisanim pravilima moguće je opisati hijerarhiju nasljeđivanja između klasa kao i organizaciju atributa unutar klase. Kako su podaci

modelovani korištenjem identifikatora, a u *GDA* upitu treba da se nađe generički identifikator, tako se i pristupanje određenoj klasi ili vrijednosti atributa klase obavlja navođenjem identifikatora.

Uloga *ResultsManagerService*-a je da dobavi rezultate korištenjem *GDA* upita i da ih u odgovarajućoj izlaznoj strukturi proslijedi prezentacionom sloju. Definisanje te strukture postiže se definisanjem klasa modela koje modeluju različite elektroenergetske elemente. Za primjer se mogu uzeti klase koje modeluju kondenzatore. Klasa *EMSLoadFlowCapacitorRecord* (slika 3.2) sa pripadajućim atributima modeluje svojstva kondenzatora. Klasa *EMSLoadFlowCapacitorResult* (slika 3.3) predstavlja enkapsulaciju klase *EMSLoadFlowCapacitorRecord*. Objekat ove klase sadrži listu objekata klase *EMSLoadFlowCapacitorRecord*. Klasa *EMSLoadFlowResult* (slika 3.4) predstavlja nadklasu za sve klase koje opisuju rezultate. Instanca ove klase prenosi se na prezentacioni sloj putem *WCF* radnog okvira.

```

[DataContract]
0 references
public class EMSLoadFlowCapacitorRecord
{
    #region Properties

    [DataMember]
    0 references
    public float VoltageLevel { get; set; }

    [DataMember]
    0 references
    public float Voltage { get; set; }
}

```

Slika 3.2 Klasa *EMSLoadFlowCapacitorRecord*

```

[DataContract]
1 reference
public class EMSLoadFlowCapacitorResult : EMSLoadFlowReportResult
{
    #region Constructors

    0 references
    public EMSLoadFlowCapacitorResult()
    {
        EMSLoadFlowReportType = EMSLoadFlowReportType.Capacitor;
    }

    #endregion Constructors

    #region Properties

    [DataMember]
    0 references
    public List<EMSLoadFlowCapacitorRecord> Records { get; set; }

    #endregion Properties
}

```

Slika 3.3 Klasa *EMSLoadFlowCapacitorResult*

```

[DataContract]
[KnownType(typeof(EMSLoadFlowGeneratorResult))]
[KnownType(typeof(EMSLoadFlowCapacitorResult))]
2 references
public class EMSLoadFlowResult : JobResult
{
    #region Constructors

    0 references
    public EMSLoadFlowResult()
    {
    }

    0 references
    public EMSLoadFlowResult(EMSLoadFlowReportType reportType)
    {
        EMSLoadFlowReportType = reportType;
    }

    #endregion Constructors

    #region Properties

    [DataMember]
    1 reference
    public EMSLoadFlowReportType EMSLoadFlowReportType { get; set; }

    #endregion Properties
}

```

Slika 3.4 Klasa *EMSLoadFlowResult*

Uloga prezentacionog sloja (*UI* komponenta) je direktna interakcija sa korisnikom, zatim slanje zahtjeva za dobavljanje rezultata ka *ResultsManagerService*-u i obrada rezultata koje servis vrati. Kod na *UI* komponenti je napisan u skladu sa *MVVM* [6] šablonom koji podrazumijeva postojanje tri komponente, a to su model podataka, prezentacioni dio (*view*) i tzv. *view-model* dio koji predstavlja spregu između modela i prezentacionog dijela.

Implementacija prezentacionog dijela napisana je upotrebom *WPF* tehnologije [7]. Prema *WPF*-u, svaka *view* klasa sastoji se od dijela koda koji je pisan u *XAML* jeziku i dijela koda koji je pisan u *C#* jeziku koji definiše ponašanje elemenata koji su prethodno definisani u *XAML*-u (tzv. *code-behind*). *XAML* je baziran na *XML* deklarativnom jeziku, koji se koristi za opis izgleda aplikacije. Sav kod napisan u *XAML*-u, predstavljen je kao hijerarhija ugnježenih elemenata, poznatijem kao stablo elemenata. U slučaju izvještaja za rezultate proračune, koji je predmet ovog rada, elementi koji figurišu u *XAML*-u su *Grid*, *TabControl*, *TabItem*, *DataGrid* i *DataGridColumn*.

View-model klase predstavljaju sponu između modela i *view* komponenti. U okviru ovih klasa nalaze se podaci neophodni za funkcionisanje prezentacionog dijela. Primjer *view-model* klase za kondenzatore prikazan je na slici 3.5.

```
public class EHSLoadFlowCapacitorViewModel
{
    #region Fields
    private EHSLoadFlowCapacitorRecord reportRecord;
    #endregion Fields
    #region Constructors
    --references
    public EHSLoadFlowCapacitorViewModel()
    {
    }
    --references
    public EHSLoadFlowCapacitorViewModel(EHSLoadFlowCapacitorRecord reportRecord)
    {
        if (reportRecord == null)
        {
            return;
        }
        this.reportRecord = reportRecord;
    }
    #endregion Constructors
    #region Properties
    --references
    public float? Voltage
    {
        get
        {
            return UnitConverterHelper.ConvertFromDPS(MeasurementType.Voltage, reportRecord.Voltage);
        }
    }
    --references
    public float? Voltagelevel
    {
        get
        {
            return UnitConverterHelper.ConvertFromDPS(MeasurementType.Voltage, reportRecord.Voltagelevel);
        }
    }
    #endregion Properties
}
```

Slika 3.5 *View-model* klasa

4. OPIS RJEŠENJA

Specifikacija metamodela implementirana je uz oslonac na *Microsoft DSL* tehnologiju [8]. Generator koda implementiran je upotrebom T4 tekstualnih šablona [9] koji referencijom dati metamodel, koriste njegove koncepte i specificiraju procese generisanja koda na osnovu tih konceptata.

3.1. ReportDSL metamodel

Specifikacija metamodela definiše tri vrste veza između elemenata modela [8] [10]:

- *EmbeddingRelationship* – veza ugrađivanja koja predstavlja takav vid relacije da su elementi na nižem nivou ugrađeni u element na višem nivou,
- *ReferenceRelationship* – veza tipa referenca predstavlja vezu gdje su elementi na nižem nivou samo referencirani od strane elementa na višem nivou,

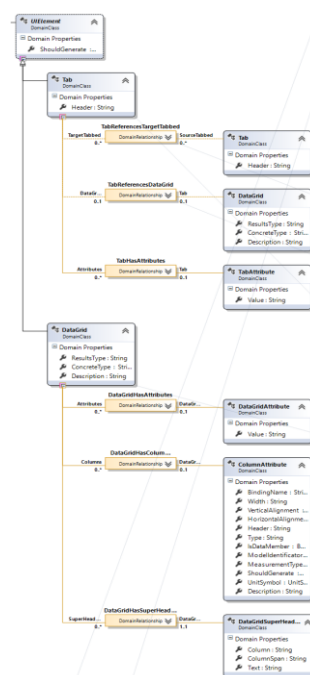
- *Inheritance* – veza nasleđivanja, gdje elementi preuzimaju osobine od elementa kojeg nasleđuju.

Svi elementi koji imaju ime, direktno ili indirektno nasleđuju klasu "*NamedElement*" koja ima atribut *Name*. Metamodel specificira apstraktnu metaklasu *UIElement* (slika 3.6) koju nasleđuju metaklase *Tab* i *DataGrid*. Ove dvije metaklase sa svojim atributima modeluju *XAML* kontrole.

Metaklasa *Tab* (slika 3.6) vezom reference ka metaklasi *Tab* specificira da *Tab* može da bude u relaciji sa nula ili više drugih *Tab*-ova. Relaciju nula ili jedan ima ka *DataGrid*-u, što znači da *Tab* može, a i ne mora da ima relaciju ka *DataGrid*-u. *Tab* ima definisanu vezu ugrađivanja "*TabHasAttributes*" ka metaklasi *TabAttribute*. Ovom vezom specificiramo da se unutar instance tipa *Tab* može naći više atributa koji je opisuju. Na osnovu vrijednosti tih atributa generišu se odgovarajući *XAML* atributi koji opisuju kontrolu *Tab*.

Metaklasa *DataGrid* (slika 3.6) posjeduje tri veze ugrađivanja. Veza ugrađivanja "*DataGridHasAttributes*" ka metaklasi *DataGridAttribute* specificira se da u okviru *DataGrid*-a može naći više atributa koji ga opisuju. Na osnovu vrijednosti ovih atributa generišu se odgovarajući *XAML* atributi. Veza ugrađivanja "*DataGridHasColumns*" specificira da jedna instanca tipa *DataGrid* sadrži nula ili više *DataGrid* kolona. Veza ugrađivanja *DataGridHasSuperHeaders* specificira da se u *DataGrid*-u može naći više instanci tipa *DataGridSuperHeader* koje modeluju nadnaslove za odgovarajuće kolone. *DataGrid* posjeduje sledeće attribute:

- *ResultsType* – naziv klase koja opisuje svojstva elektroenergetskog elementa,
- *ConcreteType* – naziv klase koja opisuje rezultate za određeni elektroenergetski element (iz enumeracije koja sadrži spisak klase modela koje se mogu instancirati),
- *Description* – opis.



Slika 3.6 Metaklase *Tab* i *DataGrid*

Metaklasa *ColumnAttribute* modeluje *DataGrid* kolonu. Posjeduje sledeće atribute:

- *BindingName* – vrijednost atributa se generiše u *XAML* kodu za istoimeni atribut,
- *IsDataMember* – označava da li će se atribut serijalizovati,
- *ModelIdentifier* – tekstualna vrijednost sa nazivom istoimenog atributa,
- *MeasurementType* – naziv mjerene fizičke veličine za koju data kolona prikazuje rezultate proračuna
- *UnitSymbol* – naziv fizičke veličine za koju data kolona prikazuje rezultate proračuna,
- *ShouldGenerate* – označava da li će se data kolona, tj atribut u modelima generisati,
- *Description* – opis kolone (klase),
- *XAML* atributi (*Width*, *VerticalAlignment*, *HorizontalAlignment*, *Header*) – čija se vrijednost direktno mapira na vrijednosti u *XAML* kodu,
- *Type* – ovim svojstvom se specificira tip podataka koji će imati vrijednosti kolone.

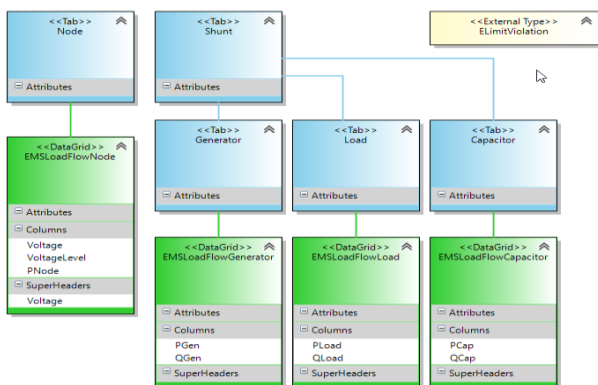
Metaklasa *SuperHeader* modeluje nadnaslov (zaglavlje) za odgovarajuće kolone. Posjeduje sledeće atribute:

- *Column* – specificira koja je prva kolona koja će biti pod datim nadnaslovom,
- *ColumnSpan* – specificira koliko kolona će se naći u okviru datog nadnaslova,
- *Text* – definiše tekst koji će imati nadnaslov.

3.1. Generisanje koda

Svaki od T4 tekstualnih šablona [9] posjeduje odgovarajuću programsku logiku pomoću koje se generiše novi tekstualni fajl koji posjeduje određenu sintaksu, a u slučaju ovog rada to je izvorni kod. U toku procesa generisanja koda, prolazi se kroz svaki od šablona i na osnovnu programske logike definiše se sadržaj generisanog izvornog koda.

Programska logika je potpomognuta ulaznim podacima koji se dobijaju sa dijagrama koji je modelovan pomoću *ReportDSL* jezika. Svaki od šablona za ulazni podatak dobija instancu modela dijagrama i u mogućnosti je da iterira kroz sve tipove elemenata sa dijagrama, pa na osnovu implementacije šablona obrađuje se element sa dijagrama za koji je šablon napisan. Primjer jednog modela izvještaja za rezultate proračuna funkcije za optimalne tokove snaga prikazan je na slici 3.7.



Slika 3.7 Primjer modela izvještaja

5. ZAKLJUČAK

Pretpostavka od koje se krenulo u implementaciju ovog rada pokazala se tačnom. Za izvještaj i izvorni kod koji je analiziran u ovom radu moguće je implementirati metamodel, odnosno generator koda, takav da se na relativno trivijalan način modeluje dati izvještaj i na osnovu tog modela generiše izvorni kod. Rezultat ove implementacije donio je mnogobrojne prednosti u razvoju softvera, a najvažnije od njih su smanjanje procenta unosa ljudske greške obzirom da se kod automatski generiše na osnovu modela, čist i pregledan kod i na kraju višestruka vremenska ušteda.

6. LITERATURA

- [1] I. Dejanović, Prilog metodama brzog razvoja softvera na bazi proširivih jezičkih specifikacija, Novi Sad: Fakultet tehničkih nauka, 2011.
- [2] I. Dejanović, Metamodel, editor modela i generator poslovnih aplikacija, Novi Sad: Fakultet tehničkih nauka, 2008.
- [3] V. Strezoski, D. Popović, N. Katić, G. Švenda, Z. Gorečan, J. Dujić / D. Bekut, Osnovne energetske funkcije za analizu, upravljanje i planiranje pogona srednjenaponskih distributivnih mreža, Kopaonik: IV skup Trendovi Razvoja: Nove tehnologije u elektrodistribuciji, 1998.
- [4] V. Strezoski, Osnovi elektroenergetike, Novi Sad: Fakultet tehničkih nauka, 2014.
- [5] IEC: 61970 403 Ed.1: Energy management system application program interface (EMS API), 2008.
- [6] M. Docs, „The MVVM Pattern,“ dostupno na: <https://docs.microsoft.com/>. [datum pristupa 03 06 2019].
- [7] M. Docs, „Windows Presentation Foundation,“ dostupno na: <https://docs.microsoft.com/>. [datum pristupa 17 06 2019].
- [8] M. Docs, „Understanding Models, Classes and Relationships,“ dostupno na: <https://docs.microsoft.com/>. [datum pristupa 03 06 2019].
- [9] M. Docs, „Code Generation and T4 Text Templates,“ dostupno na: <https://docs.microsoft.com/>. [datum pristupa 03 06 2019].
- [10] J.-P. T. S. Kelly, Domain Specific Modeling: Enabling Full Code Generation, Hoboken, New Jersey: Wiley IEEE Computer Society Pr, 2008..

Kratka biografija



Vladislav Simić rođen je u Bijeljini 02.02.1994. godine. Diplomirao je na osnovnim studijama elektrotehnike i računarstva na Fakultetu tehničkih nauka u Novom Sadu 2017. godine, a master rad je obranio 2019. godine na istom fakultetu. Kontakt: dovlasim@gmail.com

ИМПЛЕМЕНТАЦИЈА И ВИЗУАЛИЗАЦИЈА ГРАФА ТОКА КОНТРОЛЕ НА ЈЕЗИКУ PHARO**IMPLEMENTATION AND VISUALIZATION OF THE CONTROL FLOW GRAPH USING THE PHARO PROGRAMMING LANGUAGE**Светлана Ђурић, *Факултет техничких наука, Нови Сад***Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО**

Кратак садржај – У раду је пројектована хијерархија класа за моделовање графа контрола тока. Граф се преузима из синтаксног стабла произвољног *Pharo* програма и визуализује коришћењем *Mondrian* радног оквира.

Кључне речи: Граф тока контроле, апстрактно синтаксно стабло, визуализација, *Pharo*

Abstract – The paper presents a class hierarchy for control flow graph modeling. Control flow graph is created based on abstract syntax tree of a *Pharo* program. It is visualized using *Mondrian* framework.

Кључне речи: control flow graph, abstract syntax tree, visualization, *Pharo*

1. УВОД

Анализа кода може бити корисна приликом креирања и одржавања софтверских система. Анализом кода се могу изоловати графови контроле тока који представљају редослед у коме се извршавају инструкције програма. Графови контроле тока описују начине на које се контрола тока преноси кроз програм [15]. Графовима контроле тока би се могло побољшати отклањање грешака у коду. Визуализација онога што је имплементирано може знатно убрзати рад и може се корак по корак пратити шта се дешава. Графови контроле тока такође би се могли искористити и приликом утврђивања да ли ток једног програма изгледа као ток другог програма, што омогућава да се лакше препозна да ли је дошло до копирања кода.

Циљ овог рада је да се омогући издвајање и визуализација графа тока контроле из програмског кода коришћењем програмског језика *Pharo*.

2. ТЕОРИЈСКЕ ОСНОВЕ

У овој секцији је приказан *Pharo* језик као основа за имплементацију решења и описани су појмови као што је: статичка анализа кода, апстрактна синтаксна стабла, графови контроле тока и њихова намена. Описан је шаблон који је коришћен приликом имплементације овог решења, који се назива *Visitor*. Радни оквир који је послужио за визуализацију назива се *Mondrian* и појашњено је како се ради са њим.

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији ментор је била проф. др Гордана Милосављевић.

2.1 О *Pharo* језику

Pharo је савремени објектно-оријентисани језик отвореног кода. Погодан је за визуализацију, моделовање података, умрежавање, као и остале намене због бројних објектно-оријентисаних библиотека и радних оквира. Синтакса је једноставна и читљива (стаје на једну разгледницу) а објектни модел је лако прилагодљив, што одговара и почетницима и искусним програмерима.

2.2 Статичка анализа програма

Да би софтвер био квалитетан, добрих перформанси и без грешака, потребно је доста труда и времена уложити у његово пројектовање и развој. Како би се тај поступак олакшао, користи се анализа софтвера како би се откриле грешке и мане софтвера још у раним фазама. Постоје два начина за анализу софтвера: статички и динамички.

Статичка анализа софтвера се базира на анализи изворног кода. Посматрају се апстрактна и конкретна синтаксна стабла и графови контроле тока.

Динамичка анализа софтвера темељи се на подацима који се добијају у време извршавања софтвера као и интеракцијом са софтвером. Акценат овог рада је на статичкој анализи.

Програм се најбоље анализира посматрајући међукод, који је независан од коришћеног програмског језика. У овом раду биће дискутоване две форме међукода: апстрактно синтаксно стабло (АСТ) и граф тока контроле (ЦФГ). АСТ је повезано са изворним кодом, док је ЦФГ потпуно независан од језика, али се не може помоћу њега извести код. АСТ има богатији приказ у односу на ЦФГ.

Граф контроле тока се изводи из апстрактног синтаксног стабла и представља приказ изворног кода. Језички је независан, што значи да може да се изведе за било који језик. Граф контроле тока у себи садржи чворове који су повезани линијама које су усмерене од претходног ка следећем. Да би се граф тока контроле могао брже и интуитивније растумачити, добра је пракса да се изнад или испод сваког чвора уведе његово име.

2.3 *Visitor* шаблон

Шаблон *Visitor* служи да одвоји алгоритам од структуре података над којом се примењује као и да,

код сложених структура података, одреди место за имплементацију операције, која ће се извршити над елементима те структуре. Пример за такву сложену структуру може бити стабло са својим чворовима.

2.4 Mondrian

Mondrian представља језик специфичан за домен који служи за визуализацију разних врста података у облику графа. *Mondrian* омогућава интеракцију са подацима који треба да се визуализују.

3. ИМПЛЕМЕНТАЦИЈА РЕШЕЊА

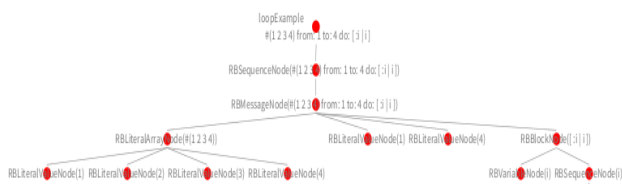
Овај рад је проширење пројекта под називом *OAK* [16]. Постојећи код за обраду графа тока контроле је проширен у оквиру овог рада и урађена је визуализација коришћењем програмског оквира *Mondrian*.

Основни принципи на којима се заснива имплементација решења јесу:

- обрађене су методе, блокови, петље, услови и једноставне структуре
- метода има своје тело
- тело може да има једну или више порука, а може да буде и празно
- блок може имати један или више аргумената; блок има своје тело; може се појавити било где.

3.1. Приказ апстрактног синтаксног стабла

Пре визуализације графа тока контроле одрађена је визуализација апстрактног синтаксног стабла. Подршка за апстрактно синтаксно стабло постоји у *Pharo* језику. Апстрактном синтаксном стаблу приступа се помоћу методе *getSource*. Прикупљају се чворови из апстрактног синтаксног стабла. Затим се апстрактном стаблу додељује посетилац. Креира се поглед и у њега се додају чворови и њихове везе. Чворови су деца из АСТ-а, а грана иде од родитеља ка деци, што је приказано на слици 1.



Слика 1 – Приказ АСТ за петљу

3.2. Опис имплементираног решења

У овој секцији је приказан дијаграм класа за моделовање графа контроле тока. Дат је опис класа.

3.2.1. Дијаграм класа за моделовање графа контроле тока

На слици 2 приказан је дијаграм класа којима се моделује граф контроле тока. Класе ће детаљније бити појашњене у наставку ове секције. Посматрајући апстрактно синтаксно стабло, лакше је изградити граф контроле тока. Са слике 2 види се да постоје два типа чворова:

- *Childish* - сложени чворови
- *Non - Childish* - прости чворови.

Сложени чворови наслеђују класу *GrdChildishNode*. Предвиђено је да постоје две врсте веза:

- претходни - следећи
- родитељ - дете

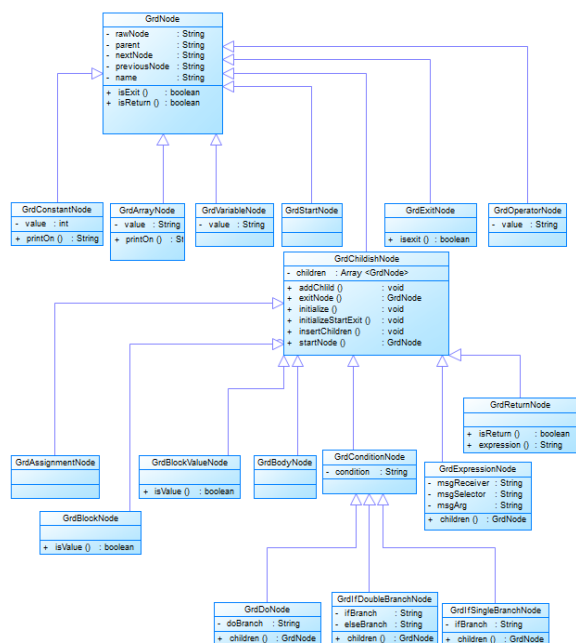
На слици 2 приказана је класа *GrdNode*, која представља чвор. Она служи за то да сваки чвор буде повезан са својим претходником и следбеником, ако их има.

Граф контроле тока почиње чворовима *GrdStartNode* и *GrdExitNode*. Они представљају улаз и излаз из графа. Ови чворови се такође користе и у оквиру сложених чворова како би било прегледније шта се дешава.

Класа *GrdChildishNode* представља сложени чвор, који у себи садржи и остале чворове. Она у себи има листу деце. У овој класи поред везе претходник - следбеник, јавља се веза родитељ - дете. Уколико је неки чвор сложен, унутар њега се приказују и његова деца и тако рекурзивно све док чвор који се посматра има деце у себи. У сложене чворове спадају чворови који представљају блокове, тело блокова, услове, чворове доделе, чворови који представљају петље, изразе и повратну вредност.

Класа *GrdBlockNode* представља блок. Класа која представља блок је наследница класе *GrdChildishNode*. Ако променљивама у блоку треба доделити вредност користи се порука *value*. Тај случај је потребно посебно обрадити и за то служи чвор кога представља класа *GrdBlockValueNode*.

Тело блока (*GrdBodyNode*), се може налазити у оквиру блока, а може и посебно, ван других чворова. Цео граф контроле тока представља чвор који је типа тело (*GrdBodyNode*). Класа која га представља је наследница класе *GrdChildishNode*.



Слика 2 - Приказ дијаграма класа за ЦФГ

Што се тиче услова, направљена је класа која представља апстракцију, *GrdConditionNode*. Она у себи садржи услов. Њене наследнице су конкретне класе за гранање. Прва је *GrdIfSingleBranchNode*, која у себи садржи грану која представља *if* грану.

Уколико се ради о чвору који има више грана, *if* и *else*, он се представља класом *GrdIfDoubleBranchNode*. Слично као код горе поменуте класе, наслеђује *GrdConditionNode*.

Петља је представљена помоћу класе *GrdDoNode*. Класа која представља обраду низова је *GrdArrayNode*. За доделу се користи класа *GrdAssignmentNode*. Чвор који је повратна вредност неке методе је представљен помоћу *GrdOperatorNode* класе, константе помоћу *GrdConstantNode*, а променљиве помоћу *GrdVariableNode*.

Уколико се ради о неком изразу, за то служи класа *GrdExpressionNode*. Класа *GrdExpressionNode* служи за обраду израза који у себи садрже бинарне операторе (*GrdOperatorNode*).

3.2.2. Посета чворовима

У овом решењу коришћен је шаблон *Visitor* који је објашњен у секцији број 2.3. У иницијализацији класе *GrdVisitor* прави се нови објекат *GrdBodyNode*, у коме ће бити смештени сви остали чворови. За сваки тип чвора направљена је посебна метода, која служи за посету истог.

У овом решењу је граф контроле тока направљен само на нивоу једног метода, где се поруке гледају само као искази. Проблем који је остао нерешен јесте повезивање позива метода, али то спада у проблеме динамичког везивања. Тај проблем би се могао решити на статичком нивоу помоћу техника као што је *type inference*. Ова техника анализира типове променљивих или израза. То значи да компајлер сам одређује тип израза, функција или вредности.

3.2.3. Визуализација графа контроле тока

За визуализацију коришћена је верзија *Pharo-a Moose Suite 6.0* јер се у њој налази све потребно за рад са *Roassal*-ом и *Mondrian*-ом. Алгоритам који се користи за обилазак графа јесте „први у дубину“. То значи да се креће од корена и да се иде дуж сваке гране докле год је то могуће.

4. ПРИКАЗ ИМПЛЕМЕНТИРАНОГ РЕШЕЊА

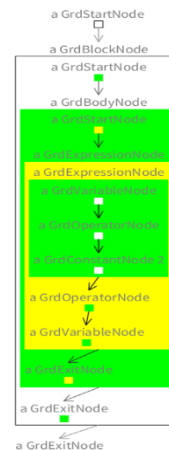
У овој секцији биће приказане слике екрана на којима се виде визуализовани графови тока контроле за примере са блоковима и условима са двоструким гранама. Слика екрана за пример блока : $[:a :b \mid a = 2 + b]$ дата је на слици 3.

На слици 3 се види граф контроле тока који се састоји од почетног чвора, чвора за блок и крајњег чвора. Чвор који представља блок има чворове приказане на слици 3. Деца од сваког чвора су увучена у односу на њега.

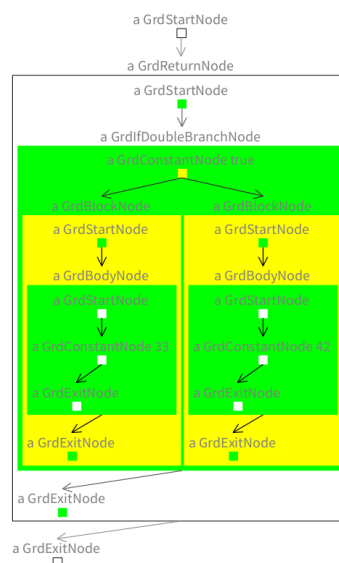
- 1) почетни чвор
- 2) тело
 - a. почетни чвор
 - b. први израз

- други израз
 - променљива *a*,
 - оператор $=$,
 - константа 2
- оператор $+$
- променљива *b*

- c. крајњи чвор
- 3) крајњи чвор



Слика 3 – Приказ графа контроле тока за блок
На слици 4 приказана је визуализација примера:
 $\wedge \text{true ifTrue: [33] ifFalse: [42]}$.



Слика 4 – Приказ графа са двоструком граном

Граф са слике 4 састоји се од почетног чвора, чвора за повратну вредност и крајњег чвора. Сва три чвора су означена белом бојом. Чвор који представља повратну вредност је сложен, и он се састоји од почетног, чвора са двоструком граном и крајњег чвора. Сва деца су му зелене боје. Са слике 4 види се да је чвор са двоструком граном сложен чвор и да се у њему налазе:

- 1) константа која представља услов (*true*)
- 2) блок који се извршава ако је услов тачан
 - a. почетни чвор
 - почетни чвор
 - константа 33
 - b. тело

- крајњи чвор
- с. крајњи чвор
- 3) блок који се извршава ако је услов нетачан
 - а. почетни чвор
 - б. тело
 - почетни чвор
 - константа 42
 - крајњи чвор
 - с. крајњи чвор

5. ЗАКЉУЧАК

У оквиру пројекта који је послужио као основа за овај рад одрађена је имплементација и визуализација графа контроле тока на програмском језику Pharo. Граф контроле тока је имплементиран и визуализован за апликације које су написане на програмском језику Pharo. Направљене су класе које представљају чворове који ће се приказивати у графу контроле тока. Написане су методе које служе за посету одговарајућим чворовима.

За просте чворове су направљене класе које описују низове, константе, променљиве и операторе који се користе у аритметичким операцијама. За сложене чворове су направљене класе које означавају доделу вредности, блокове, тело блокова, условне чворове, петље и чворове који представљају изразе. За обраду блокова је направљена додатна класа која служи за блокове са поруком *value:*. Поруком *value:* се додељује вредност променљивама које се користе у блоковима. За чворове који означавају услове су направљене класе за обраду чворова са једноструком и двоструком граном. Поруке које су обрађене су једноставне поруке и каскадне поруке.

Даљи развој подразумева рад на петљама попут: *to:do*, *to:do:by*, *timesRepeat:*. Још један вид проширења би била обрада порука попут: *ifNone:*, *ifAbsent:*, *detectIfNone:*, *at:ifAbsent:*.

Код метода се може одрадiti апстракција. То значи да се направи посетилац за методу генерално и да се та апстракција користи за сваки пример. Тако би се избегла појединачна обрада за конкретне поруке. Што се тиче визуализације, било би добро увести различите геометријске симболе и доделити им значење, тако да би корисник на први поглед одмах могао да види о чему се ради, без читања назива чвора.

ЛИТЕРАТУРА

- [1] Alexandre Bergel, Damien Cassou, Stephane Ducasse, Jannik Laval, *Deep into Pharo*, Square Bracket Associates, Switzerland, 2013
- [2] Stéphane Ducasse, Dimitris Chloupis, Nicolai Hess, and Dmitri Zagidulin, *Pharo By Example 5*, September 29, 2018
- [3] Janusz Laski, William Stanley, *Software Verification and Analysis*, Springer, 2009
- [4] Наташа Сукур, *Репрезентација тока извршавања програма дијаграмом стања, независна од улазног језика*, Мастер рад, Природно -

математички факултет, Универзитет у Новом Саду, 2016.

- [5] Visualizing Polymetric Graphs using Mondrian, <http://agilevisualization.com/AgileVisualization/Mondrian/0202-Mondrian.html>
- [6] Еволуција софтвера, <http://rti.etf.bg.ac.rs/rti/ms1es/Predavanja/ProcenaTroskova.ppt>
- [7] Оптимизација, <http://www.acs.uns.ac.rs/sites/default/files/PP9-optimizacija.pdf>
- [8] Обрасци понашања, <http://www.igordejanovic.net/courses/sok/obrasci-ponasanja.html#53>
- [9] Identifying messages , <https://ci.inria.fr/pharo-contribution/job/UpdatedPharoByExample/lastSuccessfulBuild/artifact/book-result/UnderstandingMessage/UnderstandingMessage.html>
- [10] The Pharo MOOC , <https://mooc.pharo.org/>
- [11] Smalltalk, <https://sr.wikipedia.org/sr-ec/Smalltalk>
- [12] Abstract Syntax Tree, <https://vinaytech.wordpress.com/2008/10/04/abstract-syntax-tree/>
- [13] All Pharo syntax on a postcard , <https://pharowebly.wordpress.com/2018/06/02/all-pharo-syntax-on-a-postcard/>
- [14] Стабло парсирања, https://sh.wikipedia.org/wiki/Stablo_parsiranja
- [15] Тестирање софтвера , http://etf.beastweb.org/index.php/site/download/P5_BelaKutija.pdf
- [16] OAK, <http://www.smalltalkhub.com/#!/~GordanaRakic/Oak>

Кратка биографија:



Светлана Ђурић рођена је 15.10.1995 у Зрењанину. Основне студије из области Рачунарство и аутоматика – Примењене рачунарске науке и информатика је завршила 2017. године када је уписала и мастер студије из исте области.

DIZAJN SERVERLESS WEB APLIKACIJA NA AMAZON PLATFORMI DESIGN OF SERVERLESS WEB APPLICATIONS USING AMAZON SERVICES

Aleksandar Ratkov, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu opisan je razvoj serverless web aplikacije za vođenje evidencije o godišnjim odmorima zaposlenih upotrebom Serverless radnog okvira i AWS (Amazon Web Services) servisa.

Ključne reči: AWS, Serverless, Cloud

Abstract – This paper presents an implementation of serverless web application for managing vacation days of employees using Serverless framework and services which are provided by AWS (Amazon Web Services).

Keywords: AWS, Serverless, Cloud

1. UVOD

Najnovija dostignuća u tehnologiji svakodnevno vrše pritisak na softverske programere u cilju brže implementacije novih funkcionalnosti i samom generisanju poslovne vrednosti projekta. Vreme koje je potrebno projektu da pređe svoj put od razvoja do tržišta postala je jako bitna metrika, kako zbog finansija tako i zbog straha od konkurencije. U prošlosti programeri su pored razvijanja same aplikacije trošili mnogo vremena na planiranje, implementaciju i održavanje celokupne infrastrukture na kojoj se aplikacija izvršavala.

Pojavom virtualizacije rešen je problem neiskorišćenosti hardverskih resursa što je rezultovalo značajnim smanjenjem troškova. Ovakav splet okolnosti doveo je do uspona cloud computinga, gde programeri počinju da tretiraju hardver kao uslužni program. Napredak virtualizacije bazirane na kontejnerima učinio je da se programeri postepeno prebacuju sa velikih monolitnih arhitektura na arhitekturu zasnovanu na distribuiranim mikroservisima. Ovakav iskorak prema kontejnerima i arhitekturi mikroservisa iznedrio je novu paradigmu koja je poznata kao serverless ili računarstvo bez servera.

2. CLOUD COMPUTING

Tradicionalne poslovne aplikacije su oduvek bile komplikovane i veoma skupe. Pored neophodnog hardvera i licenci za njihovo pokretanje, neophodan je i tim stručnjaka koji će voditi računa o konfiguraciji, testiranju, bezbednosti i ažuriranju istih.

Ovi problemi rešeni su upotrebom cloud computing tehnologije koja akcenat stavlja na obavljanje poslovne logike, a kreiranje i održavanje celokupne arhitekture

sistema zadatak je cloud provajdera. Postoji tri osnovna tipa cloud computinga:

- SaaS (Software as a Service)
- PaaS (Platform as a Service)
- IaaS (Infrastructure as a Service)

Softver kao usluga predstavlja model isporuke softvera na osnovu zahteva prema kojem korisnici plaćaju pristup i korišćenje funkcionalnosti softvera u obliku web aplikacija, preko Interneta. Platforma kao usluga pruža usluge neophodne za podršku celokupnog životnog ciklusa programskog proizvoda, od samog dizajniranja pa sve do isporuke aplikacija krajnjim korisnicima. Infrastruktura kao usluga predstavlja koncept iznajmljivanja hardverskih resursa kao usluge korisniku kojem je pružena mogućnost korišćenja celokupne računarske infrastrukture, uglavnom kao virtualne platforme.

2.1. Serverless Computing

Serverless computing ili jednom rečju Serverless je relativno novi koncept u softverskoj arhitekturi, koji je izazvao veliku pažnju u zadnjih par godina zahvaljujući velikom broju knjiga, projekata otvorenog koda, konferencija na ovu temu kao i ogromnim ulaganjima vodeće trojke cloud providera - Amazona, Microsofta i Googla. Najkraće rečeno, serverless arhitektura je način razvoja i pokretanja aplikacija i servisa bez potrebe za održavanjem celokupne infrastrukture. Iako sam naziv može pogrešno da sugeriše, u pitanju nije pokretanje aplikacija bez servera. Pored toga što serverless izdvaja pozadinsku infrastrukturu dalje od programera, serveri su i dalje uključeni u izvršavanje same aplikacije. Naziv serverless se koristi zbog toga što podešavanja i održavanje samog servera nije zadatak osobe koja se bavi kreiranjem sistema ili aplikacije, već je to odgovornost cloud providera. U okviru serverless-a postoje dva tipa arhitekture koja se međusobno prepliću i to su:

- BaaS (Backend as a Service)
- FaaS (Function as a Service)

Backend kao usluga je servisni model u cloud computingu koji služi kao posrednik koji pruža programerima načine da povežu svoje web i mobilne aplikacije sa cloud uslugama preko aplikacionih programerskih interfejsa i softverskih paketa. Korišćenje ovih usluga pored uštede vremena i novca u mnogome olakšavaju i rad programera koji ne moraju samostalno da implementiraju već postojeće funkcionalnosti. Funkcija kao usluga je kategorija usluga u cloud computing-u koja pruža platformu koja omogućava korisnicima da razvijaju, pokreću i upravljaju aplikativnim funkcionalnostima bez

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Branko Milosavljević, red. prof.

potrebe za izgradnjom i održavanjem celokupne serverske infrastrukture.

3. IMPLEMENTACIJA

U radu je opisana implementacija serverless web aplikacije za vođenje evidencije o godišnjim odmorima zaposlenih. Backend deo aplikacije kreiran je upotrebom Serverless radnog okvira koristeći Amazon Web Services dok je frontend deo implementiran pomoću Vue.js javascript radnog okvira.

3.1. Serverless radni okvir

Serverless radni okvir predstavlja besplatan projekat otvorenog koda koji je napisan pomoću Node.js programskog jezika, a služi za jednostavno kreiranje i deploy-ovanje serverless aplikacija koristeći command-line interface. Velika prednosti ovog radnog okvira je nezavisnost od cloud provajdera i kompatibilnost cloud funkcija u okviru različitih provajdera. Pored toga ne postoje striktna ograničenja po pitanju jezika koji se koristi za pisanje samih funkcija, već je na programeru da izabere neki od ponuđenih jezika poput Node.js, Python, Java, Go, Scala, C#. Sa Serverless radnim okvirom čitava aplikacija je definisana pomoću jednostavnog konfiguracionog fajla u yaml formatu. Jedan od preduslova za upotrebu ovog radnog okvira jeste prethodno instaliran Node.js čija verzija mora biti 6.5.0 ili novija.

3.2. Amazon Web Services

Amazon Web Services je odeljenje Amazon-a koje pruža platforme za cloud computing na zahtev pojedincima i kompanijama na bazi plaćene pretplate. Tehnologija omogućava pretplatnicima da imaju na raspolaganju virtuelnu grupu računara, dostupnu sve vreme, putem Interneta. AWS-ova verzija virtuelnih računara imitira većinu stavki pravog računara, uključujući hardver, izbor operativnih sistema, umrežavanje i unapred učitani aplikativni softver poput web servera, baze podataka itd. Troškovi za usluge korišćenja usluga AWS servisa se zasnivaju na kombinaciji upotrebe, karakteristika hardvera, OS, softvera koje je izabrao pretplatnik.

3.2.1 Cognito

Amazon Cognito je AWS-ov proizvod koji kontroliše autentifikaciju korisnika i pristup mobilnim i web aplikacijama na uređajima povezanim sa internetom. Ova usluga čuva i sinhronizuje podatke krajnjih korisnika, što omogućava programeru da se fokusira na pisanje poslovne logike aplikacije umesto na implementaciju dela koji je zadužen za autentifikaciju korisnika. Dve glavne opcije Amazon Cognita su user pools i identity pools. Prva opcija se odnosi na sakupljanje atributa profila korisnika u direktorijume koji se nazivaju grupe korisnika koje aplikacija koristi za logovanje i registraciju korisnika. Druga opcija, tzv. skup identiteta služi za obezbeđivanje AWS kredencijala korisnicima u cilju kontrolisanja pristupa AWS resursima.

3.2.2 AWS Lambda

Aws Lambda je AWS-ov servis koji omogućuje programiranje funkcija po principu plaćanja na osnovu upotrebe, bez potrebe za obezbeđivanjem memorije i

računarskih resursa za izvršavanje iste. Pisanje koda za Lambda funkciju ne zahteva poznavanje nekog novog programskog jezika već podržava postojeće programske jezike poput Node.js, Python, Java i C #, Go. Lambda funkcije su stateless, što omogućava Lambdi brzo pokretanje onoliko kopija funkcije koliko je potrebno za skaliranje kako bi se obradili svi dolazni događaji. Kao i kod svih FaaS servisa, pa i kod lambde postoje ograničenja po pitanju trajanja izvršavanja same funkcije. Trenutni limit što se tiče izvršavanja AWS Lambda funkcije je 15 minuta, nakon čega dolazi do prekida izvršavanja iste.

3.2.3 API Gateway

AWS API Gateway je servis koji se koristi za kreiranje, objavljivanje, održavanje, nadgledanje i zaštitu REST (Representational state transfer) i WebSocket API-ja. Ovaj AWS-ov servis obezbeđuje kreiranje robusnih, sigurnih i skalabilnih API-ja koji pristupaju AWS-ovim ili drugim web servisima ili podacima sačuvanim na AWS Cloud-u. U radu je API Gateway korišćen za kreiranje REST API-ja koji se sastoji od ruta, a svaka ruta je povezana sa resursom koji rukuje tom rutom. API Gateway se sastoji od funkcija, a svaka funkcija sadrži dva elementa. Prvi element je rukovaoc koji je u ovom slučaju AWS Lambda čije će izvršavanje biti aktivirano određenim događajem. Drugi element je događaj koji aktivira Lambda funkciju, u ovom slučaju HTTP poziv koji je određenog metoda i upućen je ka određenoj ruti. Kontrola pristupa samom API-ju je obezbeđena upotrebom authorizer-a koji može biti kreiran na dva načina. Prvi je upotrebom Cognito user pool-a koji vrši funkciju authorizer-a, a drugi je pisanjem nestandardne Lambda funkcije koja sadržati potrebnu logiku za autorizaciju i autentifikaciju pozivaoca zahteva.

3.2.4 DynamoDB

Amazon DynamoDB predstavlja NoSQL (not only structured query language) bazu podataka koja pruža brze performanse uz besprekornu skalabilnost. DynamoDB baza omogućava uklanjanje administrativnih opterećenja u radu i skaliranju distribuirane baze podataka, te s toga nema potrebe o brizi oko izbora hardvera, podešavanja i konfiguracije, replikaciji, krpljenju softvera ili skaliranju klastera. Takođe, DinamoDB nudi i enkripciju, čime se eliminiše složenost u zaštiti osetljivih podataka. Uz DynamoDB, moguće je kreiranje tabela baze podataka koje mogu sačuvati i vratiti bilo koju količinu podataka i pored toga opslužiti bilo koji nivo saobraćaja zahteva. DynamoDB pruža mogućnost kreiranja rezervne kopije na zahtev kao i oporavak u trenutku za AWS DynamoDB tabele. Pored ovoga omogućeno je automatsko brisanje stavki koje nisu aktuelne iz tabela kako bi se smanjila upotreba memorije kao i troškovi čuvanja podataka koji nisu više relevantni.

3.2.5 AWS Simple Email Service

Simple Email Service predstavlja AWS-ovu e-mail platformu koja pruža jednostavan i isplativ način za slanje i primanje elektronske pošte koristeći vlastite adrese e-pošte ili domene. AWS SES nudi visoku stopu isporuke e-pošte i jednostavan, trenutni pristup statistici slanja e-pošte upotrebom AWS Management Console. Takođe pruža gotova obaveštenja za uspešne i neuspešne isporuke

e-pošte kao i žalbe putem e-pošte. Amazon SES može slati povratne i žalbene poruke putem e-pošte ili putem AWS SNS (Simple Notification Service) servisa. Amazon SES nudi sandbox okruženje tj. simulator poštanskog sandučeta koji testira kako će aplikacija obraditi različite moguće situacije slanja e-pošte. Još jedna od upotreba simulatora je otkrivanje maksimalne propusnosti sistema bez probijanja dnevnog ograničenja za slanje e-pošte. Slanje e-pošte u ovom radu implementirano je upotrebom AWS Lambda funkcije i AWS SES servisa.

3.2.6 AWS Simple Storage Service

Simple Storage Service (S3) je vodeći servis ovog tipa u industriji koji nudi skalabilnost, dostupnost podataka, sigurnost i performanse kada je u pitanju skladištenje objekata. To znači da korisnici svih veličina i industrija mogu da ga koriste za čuvanje i zaštitu bilo koje količine podataka za različite slučajeve upotrebe, kao što su web sajtovi, mobilne aplikacije, oporavak i vraćanje podataka, arhiviranje, poslovne aplikacije, IoT uređaji i analitika velikih količina podataka.

Amazon S3 pruža jednostavne funkcije upravljanja, što omogućava organizaciju podataka kao i podešavanje kontrole pristupa kako bi zadovoljile specifične poslovne i organizacione zahteve. Amazon tvrdi da je S3 dizajniran sa skoro stoprocentom izdržljivošću te ne čudi podatak da se na S3 čuvaju podaci za milione aplikacija za kompanije širom sveta. S3 automatski kreira i skladišti kopije svih objekata na više sistema, što omogućava dostupnost podataka kad god je to potrebno, a sami podaci su zaštićeni od kvarova, grešaka i pretnji. Da bi se objekti skladištili na Amazon S3 neophodno je kreirati bucket. Imena S3 bucket-a se specificiraju prilikom njegovog kreiranja i globalno su jedinstvena, bez obzira na AWS region u kojem je bucket kreiran.

AWS S3 podržava različite opcije za konfigurisanje bucket-a. Moguće je konfigurisanje bucket-a za hosting web sajta, zatim konfigurisanje kako bi se upravljalo životnim ciklusom objekata u bucket-u kao i konfigurisanje bucket-a da loguje sve pristupe ka bucket-u. S3 nudi kontrolu pristupa bucket-u upotrebom korisničkih i bucket polisa koje su napisane u JSON (JavaScript Object Notation) formatu. U radu AWS-ov S3 servis je korišćen za hostovanje same web aplikacije, što je zahtevalo definisanje javnog pristupa samom bucket-u.

4. ZAKLJUČAK

Tokom godina serversko računarstvo je evoluiralo preko direktne upotrebe metalnih servera, preko virtuelizacije pa sve do koncepta računarstva bez servera (serverless). Koncept serverless računarstva omogućava brži i ekonomičniji razvoj aplikacija čemu se danas svakako teži.

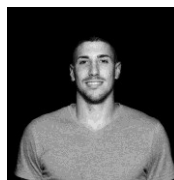
Brži razvoj aplikacije moguć je jer programeri više nemaju obavezu za implementacijom i održavanjem kompletne infrastrukture na kojoj se aplikacija izvršava. Samim tim sa finansijskog aspekta nema potrebe za nabavkom skupocenog hardvera već se koriste usluge cloud provajdera koje se plaćaju isključivo na osnovu njihove upotrebe.

Dalji pravci razvoja ovog rada bi se odnosili na automatizaciju celokupnog procesa razvoja i puštanja u produkciju aplikacije. Težilo bi se kreiranju CI/CD (Continuous Integration/Continuous Delivery) toka. Continuous Integration bi omogućio konstantu integraciju novih funkcionalnosti kao i njihovo testiranje i otkrivanje grešaka pre nego što one dospeju u produkciju. Continuous Delivery bi omogućio kraće cikluse puštanja aplikacije u produkciju koji bi bili automatizovani, što bi svakako rezultovalo smanjenjem troškova, povećanjem kvaliteta koda, srećnijim timovima koji rade na projektu kao i klijentima koji na ovaj način neuporedivo brže dobijaju nove funkcionalnosti u produkciji.

5. LITERATURA

- [1] Cloud Computing, <https://www.salesforce.com/what-is-cloud-computing/>
- [2] Cloud Computing, https://en.wikipedia.org/wiki/Cloud_computing
- [3] Serverless Computing, https://en.wikipedia.org/wiki/Serverless_computing
- [4] Serverless Computing, <https://martinfowler.com/articles/serverless.html>
- [5] Serverless Framework, https://en.wikipedia.org/wiki/Serverless_Framework
- [6] Amazon Web Services, https://en.wikipedia.org/wiki/Amazon_Web_Services
- [7] AWS Cognito, <https://docs.aws.amazon.com/cognito/latest/developer-guide/what-is-amazon-cognito.html>
- [8] AWS Lambda, <https://docs.aws.amazon.com/lambda/latest/dg/welcome.html>
- [9] AWS API Gateway, <https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html>
- [9] Serverless Computing, <https://martinfowler.com/articles/serverless.html>
- [10] AWS DynamoDB, <https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Introduction.html>
- [11] AWS SES, <https://docs.aws.amazon.com/ses/latest/DeveloperGuide/Welcome.html>
- [12] AWS S3, <https://searchaws.techtarget.com/definition/Amazon-Simple-Storage-Service-Amazon-S3>

Kratka biografija:



Aleksandar Ratkov rođen je u Beogradu 1992. god. Osnovnu školu „Boško Pal-kovljević Pinki” završio je 2007. godine. Gimnaziju „Branko Radičević ” u Staroj Pazovi završio je 2011. godine. Zvanje diplomirani inženjer elektrotehnike i računarstva stekao je 2015. godine.

SISTEM ZA PRONALAZENJE NAJKRAĆE PUTANJE IZMEĐU LOKACIJA SA TEŽINAMA PUTANJA PROMENLJIVIM U REALNOM VREMENU ZASNOVAN NA ARHITEKTURI LAMBDA**SYSTEM FOR DETECTING THE SHORTEST PATH BETWEEN MULTIPLE NODES IN REAL-TIME BASED ON LAMBDA ARCHITECTURE**

Dejan Grubišić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu predstavljen je sistem za pronalaženje najkraće putanje između više čvorova u grafu sa promenljivim težinama grana, zasnovan na arhitekturi Lambda. Moduli za paketnu obradu i obradu u realnom vremenu implementirani su u tehnologiji Spark. Realizovani su takođe i modul za vizuelizaciju, uz upotrebu Pajton biblioteke Daš, i modul za generisanje novih težina u grafu. Skladište podataka se zasniva na distribuiranom fajl sistemu Hadoop, a komunikacija između modula ostvarena je korišćenjem sistema za razmenu poruka Kafka. Sve komponente implementirane su u programskom jeziku Pajton i izvršavaju se unutar kontejnera tehnologije Docker.

Ključne reči: najkraća putanja, dinamički graf, arhitektura Lambda, Spark, veliki skupovi podataka

Abstract – In this paper, we present the system for finding the shortest path between multiple nodes in a dynamic graph. The system is based on the Lambda Architecture. Modules for batch and real-time processing are implemented in Apache Spark. Besides this, we implemented a module for visualization by using Python Dash and a module for generating new weights on edges. The storage is built on top of the Hadoop Distributed File System and communication is based on the Kafka system for exchanging messages. All components are implemented in Python and executed inside Docker containers.

Keywords: shortest path, dynamic graph, Lambda Architecture, Spark, big data

1. UVOD

Mnoge aplikacije za analizu socijalnih mreža, automatizovanje transportnih sistema kao i sam način funkcionisanja interneta, zasnivaju se na teoriji grafova. Struktura tipa grafa omogućava definisanje pojmova i određivanje veza između njih. Kompanije kao što su Gugl (eng. Google) [1] i Fejsbuk (eng. Facebook) [2] imaju u svojim sistemima milione korisnika koji svakodnevno razmenjuju poruke i generišu podatke koji se čuvaju u strukturi tipa graf. Da bi dobili potrebne informacije

o korisnicima, ove kompanije sprovode analize nad grafovima čija se struktura menja u vremenu. Obrada grafova ovih razmera zahteva posebne tehnike i umreženu hardversku infrastrukturu.

U mnogim primenama koriste se klasteri sačinjeni od računara opšte namene. Ovakvi klasteri su izuzetno fleksibilni jer omogućavaju jednostavno proširivanje novim računarima čime se povećava propusna moć. Sa druge strane, ovakvi klasteri su po pravilu znatno jeftiniji od namenskih klastera sa specijalizovanim hardverom, zbog čega se često koriste u industriji.

Usled velike kompleksnosti sistema za obradu velike količine podataka neminovno dolazi do ljudskih grešaka i potrebno je zaštititi podatke od brisanja. U slučajevima kada se čuvaju samo agregirane vrednosti podataka, može doći do problema kada postoji greška u kodu, jer je teško, a ponekad i nemoguće, rekonstruisati polazne podatke.

Način da se prevaziđe problem čuvanja podataka je da se podaci čuvaju u izvornom obliku i da se obrada pokreće nad svim podacima. Ovaj pristup garantuje preciznost i pouzdanost, ali najčešće zahteva mnogo vremena, što u primenama sa odzivom u realnom vremenu nije prihvatljivo. Da bi se dobio rezultat u željenom roku često se koriste aproksimativni algoritmi, koji omogućavaju brz odziv ali na račun preciznosti.

Zahtevi da sistem za obradu velike količine podataka bude otporan na ljudske greške i da izračunava odgovor u realnom vremenu bile su osnovni motiv za stvaranje arhitekture Lambda [3]. Kako pretraga grafa sa težinama grana promenljivim u realnom vremenu ima iste zahteve, projektovani sistem je implementiran po uzoru na arhitekturu Lambda.

2. PREGLED POSTOJEĆEG STANJA U OBLASTI

Problem pretrage grafa predstavlja jedan od osnovnih i najviše izučavanih problema teorije grafova. Ovaj problem javlja se u mnogim oblastima, zbog čega su projektovana rešenja optimizovana prema različitim vrstama grafova i prema vremenu pretrage.

Graf u opštem slučaju predstavlja strukturu čvorova koji su međusobno povezani granama. Čvorovi i grane mogu imati dodeljene osobine kao što je težina i usmerenost grana grafa, vrednosti čvorova, promenljivost ovih vrednosti u vremenu i mnoge druge. U ovom radu korišten je usmeren graf sa težinama grana promenljivim

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji je mentor bio dr Vladimir Dimitrieski, docent.

u vremenu. Ovakva struktura pogodna je za modelovanje saobraćaja, gde čvorovi predstavljaju raskrsnice, a grane puteve između njih, dok njihove težine predstavljaju vreme potrebno da se taj put pređe.

Zbog velike potrebe za brzom manipulacijom nad grafovima, razvijene su mnoge grupe algoritama. Kod statičkih algoritama (struktura grafova se ne menja u vremenu) često se uključuje korak pretprocesiranja pri čemu se dodatne informacije pridružuju čvorovima, kako bi se kasnije koristili u heuristikama pretrage [4]. Za heuristike pretrage često se koriste aproksimativni algoritmi, kada je potrebno dobiti odgovor u kratkom vremenu, pri čemu dobijeni rezultat može da odstupa u granicama zadate preciznosti.

Da bi se algoritmi dodatno ubrzali koriste se različite reprezentacije grafova kako bi se smanjio broj čvorova koji se obrađuje. Ovo se postiže redukovanjem dimenzija grafa i stvaranjem hijerarhijske strukture. Redukovana slika grafa (eng. *Contraction*) [5] se odnosi na kreiranje grana koje zamenjuju delove grafa između dva čvora, sa istom težinom putanje između njih. Na ovaj način se graf može predstaviti u više hijerarhijskih nivoa pri čemu pretraga najkraće putanje započinje na najvišem nivou hijerarhije. Kada se pronađe najkraća putanja, čvorovi na pronađenoj putanji se rastavljaju u čvorove koji ih sačinjavaju i pokreće se pretraga nad tim čvorovima. Ovo se rekurzivno nastavlja sve dok se ne dođe do najnižeg nivoa granularnosti kada se dobijaju svi čvorovi koji čine najkraću putanju.

Dinamička pretraga najkraće putanje izračunava najkraću putanju u grafu u kome se težine grana kreiraju, brišu i menjaju u vremenu. Za nalaženje najkraće putanje između svih čvorova neusmerenog bestežinskog grafa najbrži aproksimativni algoritmi su Roditi-Cvik [6] i algoritam predložen od Henzingerove, Kriningera i Nanongkaija [7] sa vremenskim kompleksnostima ažuriranja putanje respektivno $O(m \cdot n/\epsilon)$ i $O(n^{5/2})$, gde je n -broj čvorova, m -broj grana i ϵ -faktor aproksimacije. Vreme upita za oba algoritma je konstantno.

U ovom radu korišten je algoritam zasnovan na dvostranoj iterativnoj pretrazi između zadatih čvorova. Ovaj algoritam pronalazi sve putanje između zadatih čvorova u prečniku N od početnih i krajnjih čvorova ukoliko je N broj iteracija. Iako ovaj algoritam nije optimalan jer prolazi kroz sve čvorove, izabran je zbog velike skalabilnosti i mogućnosti da pretragu ograničimo na region u kome se zadati čvorovi nalaze. Kada se jednom izračunaju sve putanje, upisivanjem novih vrednosti grana u date putanje i njihovim sortiranjem, dobija se trenutna najkraća putanja.

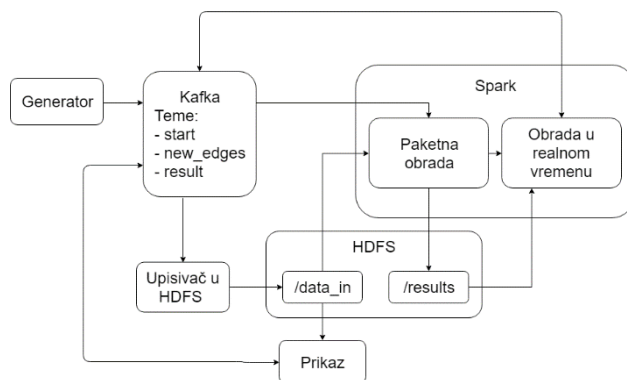
3. ARHITEKTURA SISTEMA

Cilj ovog rada je implementacija sistema za nalaženje najkraće putanje između zadatih čvorova u grafu čije su težine grana promenljive u realnom vremenu. Osnovna ideja za rešavanje zadatog problema je bila razdvajanje funkcionalnosti u dve celine:

- 1) nalaženje svih putanja između zadatih čvorova i
- 2) ažuriranje pronađenih putanja sa generisanim težinama grana i njihovo sortiranje

Nalaženje svih putanja između zadatih čvorova, zahteva intenzivno računanje i distribuiranu obradu, što odgovara onome šta izvršava modul za paketnu obradu arhitekture Lambda. Sa druge strane, zadatak koji se odnosi na ažuriranje dobijenih putanja, odgovara obradi tokova podataka i onome što izvršava modul za obradu u realnom vremenu arhitekture Lambda. Jednostavnim sortiranjem po ukupnoj težini grana ovih putanja dobija se najkraća putanja.

Ova dva modula implementirana su korišćenjem tehnologije Spark (eng. *Spark*) [8] i čine jezgro arhitekture prikazane na Slici 1.



Slika 1. Arhitektura sistema

Spark je tehnologija za obradu velikih skupova podataka opšte namene. Spark pruža mogućnost obrade podataka u realnom vremenu, koja omogućava obradu tokova podataka, skalabilnost i otpornost na otkaze. Za obradu tokova podataka definisane su takođe transformacije visokog nivoa, čime se olakšava razvoj.

Moduli projektovanog sistema povezani su posredno preko sistema za razmenu poruka Kafka (eng. *Kafka*). Kafka predstavlja sistem za razmenu poruka po principu izdavač-čitalac (eng. *publish-subscribe*) i vrlo često se koristi u ovoj oblasti. Komunikacija se odvija preko tema *start*, *new_edges* i *result* sistema Kafka. Tema *start* se koristi za prenos zadatih čvorova sa grafičkog interfejsa modula za prikaz u modul za paketnu obradu. Tema *new_edges* služi za prenos novokreiranih težina grana i tema *result* služi za slanje pronađenih putanja iz modula za obradu u realnom vremenu modulu za prikaz.

Za skladištenje početnog grafa, kao i rezultata paketne obrade koristi se distribuirani fajl sistem Hadoop (eng. *Hadoop Distributed File System, HDFS*).

Na početku obrade modul za upisivanje u *HDFS* raspoređuje izvorne podake grafa u direktorijum */data_in* (Slika 1). Ovi podaci se kasnije učitavaju u blok za paketnu obradu i prikaz grafa. Upisivač u *HDFS* ima zadatak da u toku cele obrade preuzima nove informacije o težinama grana iz generatora i ažurira njihove vrednosti u *HDFS* skladištu.

Nakon učitavanja podataka u direktorijum */data_in*, blok za prikaz generiše prikaz grafa na lokalnom serveru i pruža korisniku mogućnost da zada čvorove između kojih je potrebno izračunati najkraću putanju (Slika 2, komandna linija iznad grafika). Na prikazanom korisničkom interfejsu čvorovi su obeleženi plavom bojom, unutar žutih zvezdica je napisana težina grana,

dok je crvenom bojom obeležena najkraća putanja, čija je vrednost napisana u osmouglu u gornjem levom uglu.

Modul za prikaz implementiran je pomoću biblioteke Daš (eng. *Dash*) [9] ugrađene u programski jezik Pajton. Pomoću nje je moguće jednostavno kreiranje interfejsa prema korisniku što je pogodno koristiti zajedno sa tehnologijom Docker, koja nema ugrađene grafičke interfejse. Aplikacija za prikaz potom šalje listu zadatih čvorova brokeru modula Kafka na temu *start*, iz koje blok za paketnu obradu preuzima komande i pokreće pretragu svih putanja između zadatih čvorova.

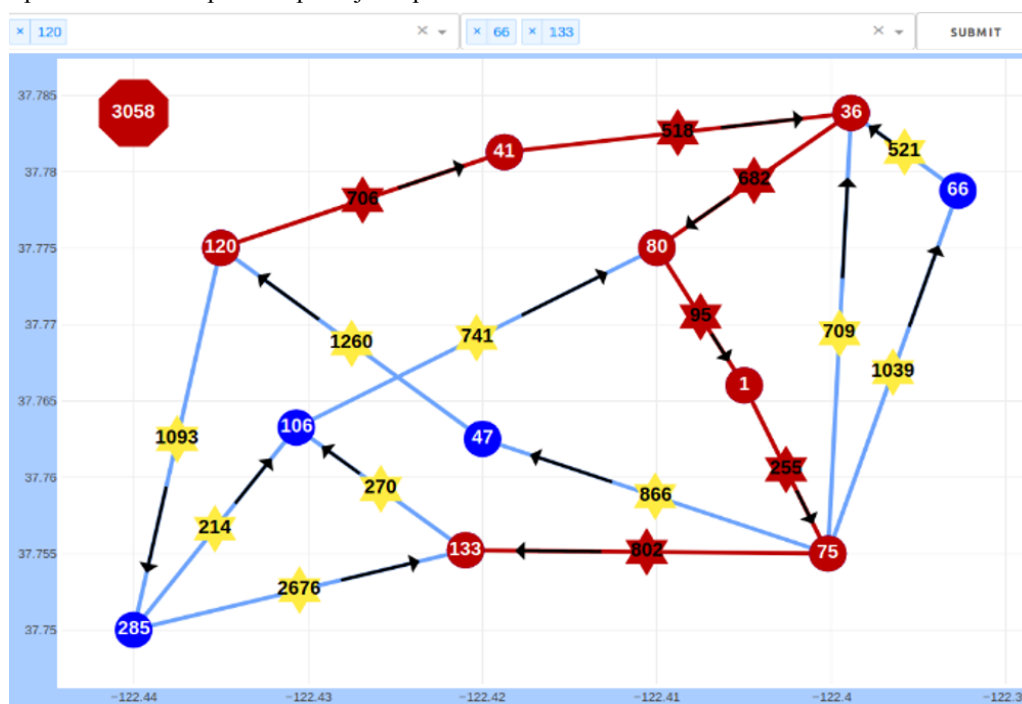
Kreiranje novih težina grana ostvareno je modulom generator. Ovaj modul šalje težine novih grana na temu *new_edges* nakon čega se ažurira prikaz i pokreće ažuriranje najkraće putanje.

Paketna obrada implementirana je iterativnim algoritmom zasnovanim na paradigmi mapiranje-redukcija (eng. *map-reduce*) tehnologije Spark. Ovaj modul kreira spisak najkraćih putanja i dobijene rezultate smešta u datoteku */results* skladišta *HDFS*. Nakon ovoga, šalje se signal modulu za obradu u realnom vremenu da preuzme rezultate paketne obrade. Po završetku, paketna obrada pokreće se ponovo i ceo proces počinje ispočetka.

Paketna obrada takođe poseduje mehanizme da prekine trenutnu obradu i krene ispočetka u slučaju da je korisnik zadao nove putanje.

Obrada u realnom vremenu zasnovana je na protočnoj verziji tehnologije Spark (eng. *Spark Streaming*). Svaki put kada modul za obradu u realnom vremenu dobije signal od modula za paketnu obradu, učitavaju se nove putanje iz datoteke */results*. Dobijene putanje se ažuriraju u zavisnosti od toka podataka novih težina grana, počevši od trenutka pokretanja paketne obrade poslate od strane generatora. Preuzimanje novih težina implementirano je u mikro-paketnom maniru (eng. *Micro-batching*) čime se omogućava paralelno ažuriranje putanja. Putanje se potom sortiraju i šalju modulu za prikaz, preko Kafka brokera na temi *result*.

Sistem za prikaz preuzima i prikazuje najkraću putanju (Slika 2) i lokalno ažurira vrednosti težina grana u grafu dobijenih preko Kafka brokera na temi *new_edges*. Ovakav postupak omogućava da vrednosti novih putanja odmah budu prikazane na grafiku, dok dobijanje najkraće putanje dolazi sa zakašnjenjem od modula za obradu u realnom vremenu.



Slika 2. Prikaz grafičkog interfejsa aplikacije

Komponente sistema realizovane su u okviru kontejnera Docker (eng. *Docker*) [10] pokrenutih u okviru operativnog sistema Linuks (eng. *Linux*).

4. ALGORITAM ZA NALAŽENJE NAJKRAĆE PUTANJE

Algoritam za nalaženje svih putanja zasniva se na dvostranoj iterativnoj pretrazi od početnih do krajnjih čvorova. Tokom pretrage prolazi se kroz sve čvorove u prečniku N , gde je N broj iteracija.

Iako ovaj algoritam nije optimalan jer prolazi kroz sve čvorove ovaj pristup odabran je jer se lako skalira i

moguće je ograničiti pretragu na region u kom se nalaze početni i krajnji čvorovi.

Kada su jednom izračunate sve putanje u regionu prosto ažuriranje tih putanja sa novim težinama grana dovodi do nalaženja najkraće putanje, što može biti urađeno u vremenu $O(\text{NewEdges} \cdot \text{Paths} / p)$, gde *NewEdges* predstavlja broj ažuriranih grana, *Paths* označava broj putanja i p je broj procesora.

Algoritam za nalaženje svih putanja se sastoji iz 5 koraka i pseudokod prikazan je u Listingu 1.

```

for i in range(steps):
    neighbors_list = map(graph)
    neighbors_list = reduceNeighbors(neighbors_list)
    graph = join(graph, neighbors_list)
    graph = updateGraph(graph)
    graph = sortPaths(graph)

results = findConnectedNodes(graph)

```

Listing 1. Algoritam za nalaženje svih putanja

Graf je predstavljen listom čvorova pri čemu svaki čvor ima listu svojih suseda. Na početku programa se svakom čvoru dodaje par (*udaljenost*, *putanja*) od početnih i do krajnjih čvorova. Za čvorove između kojih se traže putanje dodeljuje se vrednost (0, ""), dok se za ostale postavlja (*inf*, ""), gde je *inf* najveći broj u sistemu.

U prvom koraku funkcija *map* određuje listu susednih čvorova za sve čvorove koji poseduju putanju od početnih ili do krajnjih čvorova, odnosno poseduju par čija je *udaljenost* različita od *inf*. Potom se kreira putanja za svaki susedni čvor, dodavanjem identifikatora datog čvora i težina između datog i susednog čvora na postojeću putanju datog čvora. Udaljenost do susednih čvorova dobija se kao zbir udaljenosti datog čvora i težine grane između datog i susednog čvora.

Kako je moguće da jedan čvor bude susedan sa više početnih čvorova funkcija *reduceNeighbors* vrši spajanje podataka o putanjama u liste putanja za svaki susedni čvor iz liste. Ovi podaci se u sledećem koraku pridodaju polaznim podacima funkcijom *join*. Funkcija *updateGraph* koristi se za ažuriranje susednih čvorova u polaznom grafu novim putanjama prema početnim i krajnjim čvorovima kreiranim u prethodnim koracima. Nakon toga se date putanje sortiraju od najmanje do najveće udaljenosti funkcijom *sortPaths*.

Nakon zadatog broja iteracija nalaze se čvorovi koji imaju putanje povezane i sa početnim i sa krajnjim čvorovima, koji zajedno čine kompletnu putanju. Čvorovi sa povezanim putanjama se sortiraju prema ukupnoj težini grana i uzima se najkraća od njih. Na ovaj način dobija se najkraća putanja između zadatih čvorova, a sve jedinstvene putanje prosleđuju se modulu za obradu u realnom vremenu koji ih ažurira novokreiranim težinama grana i traži trenutnu najkraću putanju.

5. ZAKLJUČAK

U ovom radu opisan je sistem za nalaženje najkraće putanje u grafu velikih dimenzija sa težinama grana promenljivim u realnom vremenu. Kako bi se ovaj problem efikasno rešio podeljen je u dva koraka. U prvom koraku se nalaze sve putanje između zadatih čvorova, dok se u drugom koraku date putanje ažuriraju sa novokreiranim težinama grana i sortiraju kako bi se pronašla najkraća putanja.

Nalaženje svih putanja predstavlja računarski intenzivan posao, koji može lako da se skalira, što odgovara onome što radi modul za paketnu obradu arhitekture Lambda. Drugi korak, za ažuriranje putanja, predstavlja jednostavniji problem od polaznog i može biti obuhvaćen

modulom za obradu u realnom vremenu arhitekture Lambda.

Razlika implementiranog sistema i arhitekture Lambda je u tome što modul za obradu u realnom vremenu koristi rezultate paketne obrade i počinje nakon nje. U mnogim primenama kao što su regulacija saobraćaja ovo ne predstavlja problem, jer se kreiranje novih ulica, što je ekvivalentno dodavanju grana u grafu, ne događa brzo, u odnosu na promenu gustine saobraćaja koja odgovara promeni težina postojećih grana.

Dalji napredak mogao bi biti ostvaren hijerarhijskom reprezentacijom strukture grafa, pri čemu bi se svaki sloj dobijao spajanjem određenog broja susednih čvorova prethodnog sloja. Pretraga bi započinjala od najvišeg nivoa hijerarhije i prelazila bi na niži nivo, kada se pronađe najkraća putanja. Na ovaj način, značajno bi redukovali graf, čime bi se mogle detektovati putanje i na udaljenosti većoj od $2N$, gde je N broj koraka.

6. LITERATURA

- [1] „GoogleAI Graph Mining,“ <https://ai.google/research/teams/algorithms-optimization/graph-mining/>. [8. 7. 2019.]
- [2] A. Ching et al., „One Trillion Edges: Graph Processing at Facebook-Scale“, International Conference on Very Large Data Bases, 2015.
- [3] N. Marz i J. Warren, Big Data Principles and Best Practices of Scalable Realtime Data Systems, Manning Publications, 2015.
- [4] P. Hart, N. Nilsson i B. Raphael, „A Formal Basis for the Heuristic Determination of Minimum Cost Paths“. Computer Science and cybernetics, 1968.
- [5] R. Geisberger i P. Sanders, „Exact routing in large road networks using contraction hierarchies“, u *Transportation Science*, 2012.
- [6] L. Roditty i U. Zwick, „Dynamic Approximate All-Pairs Shortest Paths in Undirected Graphs“, SIAM Journal on Computing, 2012.
- [7] M. Henzinger, S. Krinninger i D. Nanongkai, „Dynamic Approximate All-Pairs Shortest Paths: Breaking the $O(mn)$ Barrier and Derandomization“, Annual Symposium on Foundations of Computer Science, 2013.
- [8] „Apache Spark,“ <https://spark.apache.org/>, [24. 6. 2019.].
- [9] „Dash User Guide,“ <https://dash.plot.ly/>, [24. 6. 2019.].
- [10] „Docker,“ <https://www.docker.com/>, [24. 6. 2019.].

Kratka biografija:



Dejan Grubišić rođen je 1996. godine u Novom Sadu, Srbija. Fakultet tehničkih nauka upisao je 2014. Bečelorski rad na usmerenju za „Embeded sisteme i algoritme“ odbranio je 2018. godine. Master rad na usmerenju za „Računarstvo visokih performansi“ odbranio je 2019. godine.

METODE ISPITIVANJA I GARANCIJE KVALITETA U OBLASTI SOFTVERSKIH ALATA
TESTING METHODS AND QUALITY ASSURANCE IN THE FIELD OF SOFTWARE TOOLSMaja Milišić, Željen Trpovski, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – U radu je prikazan jedan način automatskog ispitivanja SCS (eng. SoundClear Studio) razvojnog okruženja zasnovanog na regresivnom izvršavanju ispitnih slučajeva na sistemskom novou. Akcenat je stavljen na ispitivanje dve glavne komponente aplikacije: serverski deo, koji radi kao alat iz komandne linije zasnovan na gRPC okviru, napisan u programskom jeziku C# i GUI klijentu, implementiranom u programskom jeziku Java.

Ključne reči: ispitivanje, ispitni slučajevi, SCS, IDE, DUT, API

Abstract – The paper presents one method of automated testing of the SCS (eng. SoundClear Studio) development environment based on regressive execution of test cases on the system level. The emphasis is placed on testing the two main components of the application: the server part, which works as a command line tool based on the gRPC framework, written in the programming language C# and the GUI client, implemented in the Java programming language.

Keywords: testing, test cases, SCS, IDE, DUT, API

1. UVOD

Integrirana razvojna okruženja (eng. Integrated Development Environment - IDE) najčešće predstavljaju jednu aplikaciju u koju su ugrađeni: razvojni alati, grafička korisnička sprega, komunikacioni protokoli i sve što je potrebno za rad sa savremenim realnim sistemima. Primer ovakve aplikacije je SCS (eng. SoundClear Studio). Ova aplikacija pruža širok opseg mogućnosti: za pisanje programa, njegovo spuštanje na sisteme, za kontrolisano izvršavanje programa, otklanjanje uočenih grešaka i uopšte, za razvoj onoga što će namenski sistemi raditi.

Karakteristika ovog okruženja je da je za njegovo razvijanje izabran agilan pristup – strategija koja označava paralelan rad tima za razvoj i tima za ispitivanje. Cilj pomenutog pristupa je da se razvojno okruženje kao proizvod implementira i isporučuje kroz više nezavisnih međuverzija, gde će svaka od njih posedovati određen skup unapred definisanih funkcionalnosti, ali i zahteva koje su klijenti u međuvremenu definisali. Shodno tome, ispitivanje i povratne informacije moraju se takođe obavljati kroz više nezavisnih iteracija, inkrementalno, onako kako razvoj okruženja napreduje.

Rad je nastao kao rezultat ispitivanja SCS razvojnog okruženja, softverskog alata koji se razvijao u Naučno-

istraživačkom institutu “RT-RK” [1] i kompaniji Cirrus Logic [2]. To je novo razvojno okruženje koje predstavlja veći skup pojedinačnih alata namenjenih svim Cirrus-ovim klijentima. Ispitivanje koje treba obaviti, na v1.6.1.0 verziji SCS aplikacije, biće zasnovano na primeni različitih tipova ispitnih slučajeva, jer na taj način garancija da je sistem pouzdan je sve veća. Ispitivanje se u najvećem broju slučajeva sprovodi na udaljenim platformama [3].

Pored toga, potrebno je rešiti i problem ispitivanja razvojnog okruženja na različitim operativnim sistemima. Cilj je da se proces ispitivanja u potpunosti automatizuje, da bude što pouzdaniji, prilagodljiv aktuelnim operativnim sistemima, ali i nezavisan u odnosu na prekid komunikacije između platforme i ispitivača. Automatizacijom procesa ispitivanja se postiže značajna ušteda vremena i resursa. Automatski ispitni slučajevi se izvršavaju brže, zahtevaju manje resursa i manje obučanih ljudi. Ispitno okruženje i ispitni slučajevi su realizovani u Python programskom jeziku, skriptnom jeziku visokog nivoa u kom je čitanje i pisanje koda veoma jednostavno i lako za učenje.

2. ANALIZA RAZVOJNOG OKRUŽENJA

SCS čini centralno mesto razvojnog okruženja predstavljajući uređaj nad kojim se sprovodi automatizovan proces testiranja, (eng. Device Under Test - DUT). Kontrola između SCS aplikacije i sistema ostvaruje se pomoću Studio link-a, podsistema zaduženog za komunikaciju. Arhitektura okruženja prikazana je na slici 1. Kao alat za evaluaciju, programiranje i korisničko konfiguirisanje razvojnih ploča, pruža sledeće mogućnosti:

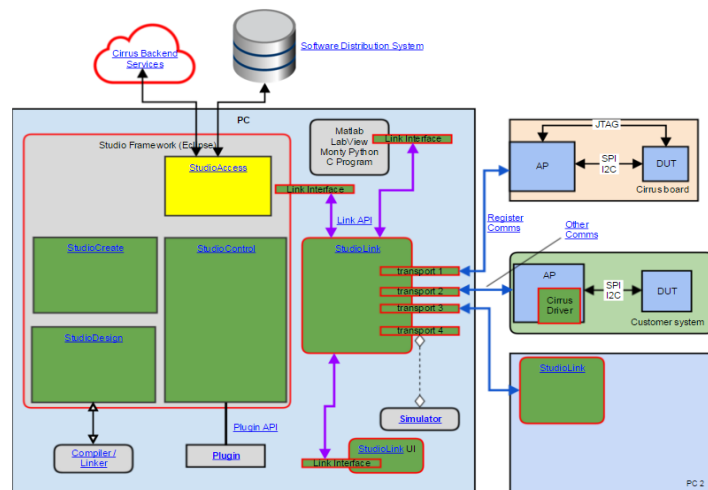
- Instalaciju na različite operativne sisteme
- Komunikaciju sa svim starim i novim DSP procesorima kompanije Cirrus Logic
- Povezivanje računara i ploča sa procesorima preko različitih fizičkih medija
- Različite korisničke sprege: Python, Matlab, LabView
- Pisanje programa, njihovo povezivanje i spuštanje na DSP
- Kontrolisano izvršavanje programa

Jedan od mnogobrojnih podržanih čipova koji je moguće konfiguirati pomoću SCS-a jeste CS46L41, poznatiji pod imenom Gray.

Gray jeste audio kodek male snage sa USB interfejsom. Razvila ga je američka kompanija Cirrus Logic koja je prepoznatljiva po tome što, pored samog čipa, klijentima nudi celokupno hardversko i softversko rešenje.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Željen Trpovski.



Slika 1. *Arhitektura razvojnog okruženja*

To znači da je kodek integrisan u okruženje i čini jednu od karika u lancu blokova obrade. S toga se javlja potreba za postojanjem alata koji će konfiguraciju čitave platforme učiniti jednostavnom i pristupačnom. Ima najveću primenu u audio adapterima (BAA) i slušalicama (BHS).

Ispitivanjem se moraju prevazići problemi geografske udaljenosti (ispitne mašine se nalaze na različitim kontinentima) i različitih operativnih sistema (Windows 7, Windows 10, Mac i Linux). Samim ti, SCS aplikacija instalirana na ispitnoj mašini mora biti lako dostupna, a sa druge strane ispitno okruženje mora biti kompatibilno i ne sme zahtevati dodatnu programsku podršku za rad.

Ispitivanje SCS razvojnog okruženja se sastoji iz nekoliko zahteva i to:

- Ispitivanje instalacije
- Ispitivanje klijentskog dela okruženja (Studio IDE)
- Ispitivanje serverskog dela okruženja (Studio link)

Ispitivanje instalacije se odnosi na proveru da li je SCS razvojno okruženje ispravno instalirano na mašini gde se obavlja ispitivanje. Nakon instalacije SCS-a, pri inicijalnom pokretanju aplikacije potrebno je instalirati BSP pakete, koji omogućavaju aplikaciji da posredstvom linka komunicira sa željenim Cirrus Logic uređajima, bilo da je u pitanju realan ili virtuelni sistem. Odgovarajući BSP se isporučuju zajedno sa aplikacijom.

2.1. Studio IDE

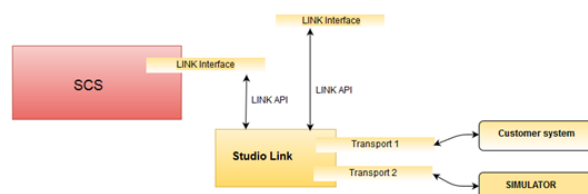
Ispitivanje klijentskog dela počinje od komponente navigator. Ova komponenta otkriva koji su sistemi prisutni na mreži, u našem slučaju ploče sa podržanim čipovima. Nakon toga se proverava da li je moguće povezivanje i njihova kontrola upisom i čitanjem vrednosti iz registara.

Ispitivanje se nastavlja pregledom istorije kontrolnih akcija, gde je u listi akcija snimljena svaka prethodna akcija korisnika, vreme kada se desila kontrola nad registrima, itd. Pored toga, ispituju se i razni meniji, tulbarovi i svi ostali delovi dostupni klijentu. Pomenute komponente i funkcionalnosti ispitane su u dve dimenzije i to pri radu sa simuliranim i realnim sistemima tj. pločama.

2.2. Studio link

Studio link je komponenta razvojnog okruženja koja se isporučuje zajedno sa evaluacionim pločama i pruža jedinstveni interaktivni interfejs koji omogućava kontrolu i pregled podešavanja uređaja. Ispitivanje serverskog dela, Studio linka, treba da proveri funkcionalnosti koje se odnose na: ispravno otkrivanje ploča na mreži, komunikaciju i kontrolu ploča, čuvanje istorije poslatih komandi i njihovih povratnih vrednosti, vreme opsluživanja klijentskih zahteva i slično.

Višestruke klijentske aplikacije mogu se povezati i koristiti dostupan hardver koji je istovremeno povezan na mrežu. Studio link API omogućava pisanje programa koji komuniciraju sa hardverom preko Studio link-a. Svi delovi SCS-a komuniciraju sa sistemima na ovaj način. API je realizovan preko Studio link interfejsa, tako što koristi Studio link protokol za slanje komandi aplikaciji. Postoji podrška za SCS Client API u više programskih jezika, što daje mogućnost komunikacije SCS-a sa komponentama koje su realizovane u različitim programskim jezicima. Trenutno SCS podržava Java i Python API. Sa druge strane link komunicira sa sistemom (hardverom) preko Studio link transporta, kojih takođe ima više u zavisnosti od vrste sistema sa kojim se komunicira. Obostrana komunikacija je prikazana na slici 2.



Slika 2. *Prikaz komunikacije aplikacije i sistema*

3. REALIZACIJA ISPITNOG OKRUŽENJA

Ispitivanje SCS razvojnog okruženja realizovano je na sistemskom nivou, što znači da se razvojno okruženje posmatra kao integrisan sistem. Potrebno je sprovesti ispitivanje na različitim platformama, pa ispitno okruženje mora biti realizovano tako da se identično ponaša na svakoj od njih.

To je razlog za implementaciju okruženja na bazi klijent-server arhitekture. Upotreba skriptnog programskog jezika Python pokazala se kao odličan izbor pri realizaciji iz više razloga. Njegove osnovne prednost su da se lako i brzo uči, dostupan je bez posebnih komercijalnih dozvola, pruža podršku za objektnu metodologiju i što je najvažnije nezavisan je od platforme na kojoj se prevedeni program izvršava.

Ispitno okruženje sadrži više različitih modula napisanih u Python-u. Postoje moduli koji služe za zapis i čuvanje poziva svih funkcija, svih akcija i događaja u tekstualne datoteke (tako da se posle izvršavanja ispitnog slučaja one mogu pregledati), moduli koji razrešavaju o kojoj ispitnoj mašini se radi i svim putanjama na njoj, modul koji pruža podršku za pokretanje Squish alata, modul u kome je realizovana podrška za automatizaciju koraka u ispitnim slučajevima, kao i modul koji obezbeđuje i održava komunikaciju između ispitne i klijent mašine.

Glavne softverske komponente okruženja su:

- Client-server
- Squish
- Test case i test suite
- Machine specific

Client-server paket se sastoji od server i klijent API-ja. Server se pokreće na ispitnoj mašini na kojoj se izvršavaju automatizovani ispitni slučaj. Može biti mašina sa bilo kojom podržanom platformom (Windows 7, Windows 10 ili Mac). Okruženje je implementirano tako da je prvo potrebno pokrenuti server API, tada je server spreman i čeka zahteve od klijent host-a. Klijent strana je strana na kojoj se pokreće RT-Executor i to mora biti isključivo Windows mašina. Klijent API pokreće Intent+ skripta koju poziva RT-Executor prilikom inicijalizacije ispitnih slučajeva.

Squish paket čini Squish API koji omogućava pokretanje i izvršavanje automatizovanih ispitnih skripti implementiranih u Squish-u. Glavne komponente API-ja su Squish server, runner i metoda za razrešenje statusa ispitnog slučaja. Izvršavanje ispitnog slučaja započinje pozivom metode *runSquishTestCase(squishTest)* koja prvo pokreće server, izvršava ispitni slučaj, postavlja status i zaustavlja

server. Poziva se u main funkciji ispitne skripte na server ispitnoj mašini.

Test case i test suite paketi služe za sumiranje rezultata izvršenih ispitnih slučajeva. Sadrže metode za ispisivanje i generisanje log poruka i izveštaja na predefinisanim putanjama. Pored logova ispitnih slučajeva generišu se i log poruke čitavog okruženja.

Metode implementirane u ovom paketu koriste osobine takođe implementirane *Logger* klase. Ovaj paket je značajan jer pored ispitnih slučajeva implementiranih u Squish-u moguće je izvršavati i čiste Python ispitne skripte.

Machine specific paket sadrži xml u kom su pobrojane sve ispitne mašine na kojima se izvršavaju ispitni slučajevi. Ova skripta se koristi za uspostavljanje komunikacije i putanja između klijenta i servera na automatizovan način.

Na klijent strani, u sklopu RT-Executor-a u *user_spec_config.py* skripti se navodi putanja do klijenta i ime ili IP adresa ispitne mašine na kojoj će se vršiti ispitivanje.

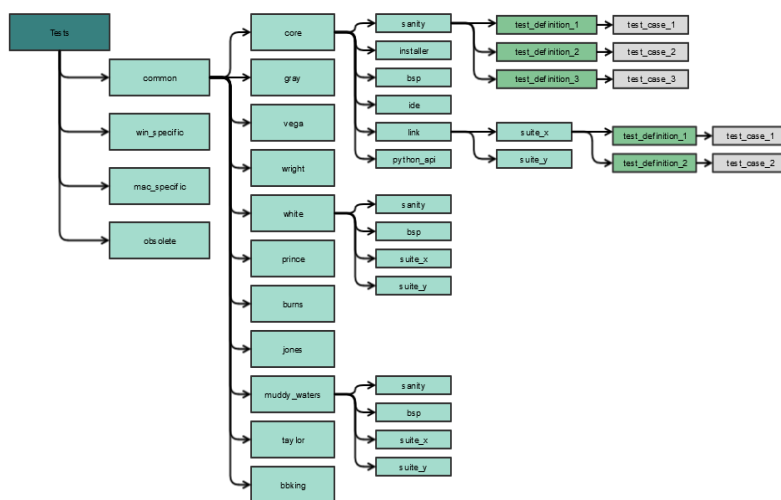
Na server strani, machine specific metode vrše parsiranje xml-a i na osnovu poslatog imena ispitne mašine znaju se svi ostali potrebni podaci o računaru. Xml polje svakog računara sadrži predefinisani skup putanja koje moraju postojati na svakom računaru na kom se izvršavaju ispitni slučajevi.

3.1. Tipovi ispitnih slučajeva

Ispitivi slučajevi se mogu podeliti u tri veće celine: ispitivanje instalacije, ispitivanje serverskog dela (Studio link) i ispitivanje klijentskog dela (Studio IDE).

Najveću grupu ispitnih slučajeva čini ispitivanje IDE-a tj. korisničkog interfejsa. Proverava se ispravnost na korisničkom nivou, kucanjem po tastaturi ili klikom miša se ispituje da li se dobijaju očekivani rezultati na akcije koje su preduzete.

Ispitivanje SCS alata se najvećim delom zasniva na GUI ispitivanju. Podela i struktura realizovanih ispitnih slučajeva prikazana je na slici 3.



Slika 3. Struktura ispitnih slučajeva

Gray, Vega, White testovi – su grupe ispitnih slučajeva za određene SCS projekte. SCS podržava više razvojnih ploča, i samim tim poseduje različite funkcionalnosti u zavisnosti od želja i potreba klijenta. Reed ploča se koristi kao podrazumevana za ispitivanje core funkcionalnosti. Na ovoj ploči se ispituju svojstva alata koja su zajednička za sve razvojne ploče.

U situacijama kada klijent zahteva ispitivanje aplikacije za potrebe Gray projekta tada se izvršavaju core + gray testovi (koji su karakteristični za funkcionalnosti tog projekta).

3.2. Implementacija ispitnih slučajeva na sistemskom novou

Automatske ispitne procedure u ovom radu implementirane su u Python programskom jeziku. Python ispitne skripte sadrže niz komandi/koraka preko kojih se, uz pomoć API biblioteka, šalju poruke uređajima koji čine sklop ispitnog okruženja.

Svaka ispitna skripta nosi naziv *test.py* i sadrži sledeće elemente:

1. Zaglavlje, u kojem se nalaze informacije o ispitnom slučaju,
2. Uključivanje biblioteka i konstanti koje su korišćene u ispitnoj proceduri,
3. Glavnu funkciju, *main()*, u kojoj je sadržana logika čitavog ispitnog slučaja,
4. *executeSteps()* metoda koja omogućava izvršavanje ispitnog slučaja.

4. PRIKAZ REZULTATA ZA VERZIJU v1.6.1.0

U okviru SCS projekta najviše je zastupljeno regresivno ispitivanje. Svaki put kada se pojavi nova verzije SCS alata potrebno je izvršiti regresivno ispitivanje u predviđenom vremenskom roku. Na taj način se proveravaju funkcionalnosti novih komponenti i proverava se da nije došlo do promena u načinu rada nekih funkcija koje nisu bile obuhvaćene izmenama.

Upravo se ovde ogleda značaj automatizacije procesa ispitivanja jer je u tom slučaju ušteda vremena i resursa znatno veća. Regresivno ispitivanje je najčešće praćeno istraživačkim ispitivanjem, koje se oslanja na iskustvo i sposobnost pojedinca da pronađe nepravilnost u radu sistema. Defekti koji se tom prilikom otkriju, evidentiraju se i prijavljuju nadležnom timu kako bi što pre bili otklonjeni.

Ispitivanja počinu izvršavanjem sanity grupe ispitnih slučajeva na sva tri operativna sistema. Ukoliko je sanity ispitivanje uspešno izvršeno, prelazi se na regresivno ispitivanje svih ostalih komponenti sistema.

U zavisnosti od zahteva klijenta regresivno ispitivanje se sprovodi ili u celosti ili samo nad zavisnim delovima sistema na koje promene mogu uticati. Isto tako ispitivanje se može sprovesti samo na nekoj od podržanih platformi.

Nakon analize zahteva za verziju v1.6.1.0, kreiranja ispitnih slučajeva, njihove automatizacije i izvršavanja na Windows 10 i Mac platformi, dobijeni su sledeći rezultati:

Tabela 6.1. Sanity ispitni izveštaj sa statistikom

Test plan	failed	passed	pending	Not tested	Not applicable	delayed	Total
SCS_sanity_win10	0	80	0	0	0	0	80
SCS_sanity_win7	0	80	0	0	0	0	80
SCS_sanity_mac	0	80	0	0	0	0	80

Tabela 6.2. Regresivni ispitni slučajevi sa statistikom

Test plan	failed	passed	pending	Not tested	Not applicable	delayed	Total
SCS_v1.6.1.0_win10	6	400	0	0	0	0	406
SCS_v1.6.1.0_mac	5	452	0	0	0	0	457

5. ZAKLJUČAK

U radu je predstavljena uspešna realizacija automatskog ispitivanja SCS razvojnog okruženja. Ostvarena je nezavisnost ispitnog okruženja u odnosu na različite platforme na kojima se vrši ispitivanje. Ispitno okruženje je jednostavno za upotrebu i daje mogućnost odlične podrške za nastavak ispitivanja svih novih verzija SCS alata.

6. LITERATURA

- [1] RT-RK, "RT-RK Institute for Computer Based Systems", 2019, [Online]. Available: <http://www.rt-rk.com/>. [Accessed: 27.05.2019.]
- [2] Cirrus Logic, "Cirrus Logic, Inc.", 2019, [Online]. Available: <https://www.cirrus.com/>. [Accessed: 27.05.2019.]
- [3] M. Tankosić, U. Mičić, Đ. Miljković, I. Kaštelan, "Automatizacija ispitivanja integrisanog okruženja za DSP na principu crne kutije", Novi Sad, 2016.

Kratka biografija:



Maja Milišić rođena je u Šapcu 1992. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Telekomunikacije i obrada signala odbranila je 2019. god. kontakt: milisic.maja@yahoo.com



Željko Trpovski rođen je u Rijeci 1957. godine. Doktorirao je na Fakultetu tehničkih nauka 1998. god. Oblast interesovanja su telekomunikacije i obrada signala.

UPRAVLJANJE SAVREMENIM KOMPRESORSKM SISTEMIMA ZA TRANSPORT INDUSTRIJSKIH GASOVA**CONTROL OF MODERN COMPRESSOR SYSTEMS FOR TRANSPORT OF INDUSTRIAL GASSES**Luka Janičić, Boris Dumnić, *Fakultet tehničkih nauka, Novi Sad***Oblast –ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – U ovom radu je opisan način realizacije upravljačkog sistema za kompresore velike snage, koji služe za transport industrijskih gasova. Dat je primer realizacije upravljanja kompresorskim sistemom za transport hlora. Prikazan je proces pripreme relevantne dokumentacije za proizvodnju ormara automatike, opis opreme koja je implementirana kao i funkcionalna specifikacija softvera, koji služi za upravljanje pogonom.

Ključne reči: upravljanje, kompresor, industrijski gasovi

Abstract – This paper describes realisation of control system for high power compressors, which are used for transport of industrial gasses. In paper is given example of compressor control system for transport of chlorine. Process of preparation of relevant documentation for production of control system cabinet, description of equipment that is implemented as well as functional specification of software for compressor control are presented.

Keywords: control, compressor, industrial gasses

1. UVOD

U mnogim procesnim industrijama kao što su hemijska i petrohemijska, u pogonima za transport tečnog naftnog gasa, u rafinerijama i sl. potrebno je da se određeni gas transportuje sa jedne lokacije na drugu.

Da bi se ovo postiglo, potrebno je da gas bude na adekvatnom pritisku kao i da se ostvari adekvatan protok. Oba prethodno navedena ključna parametra, vezana za transport industrijskih gasova, postižu se primenom kompresorskih sistema.

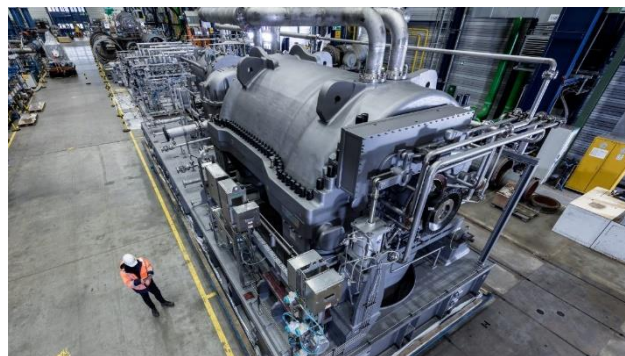
Kompresor u osnovi predstavlja sklop koji služi da podigne pritisak određenog gasa u cevi, tako što mu smanji zapreminu ili tako što mu prenese kinetičku energiju rotacije mešača (*eng. impeller*) [1].

Najveća primena kompresora velike snage je u industriji nafte i gasa. Jedna platforma za eksploataciju tečnog naftnog gasa (TNG) je vrlo kompleksan sistem sačinjen od velikog broja različite opreme koja tako integrisana zahteva kompleksna rešenja automatizacije.

U zavisnosti od aplikacije postoji veliki broj različitih vrsta kompresora koji se primenjuju u industriji nafte i gasa. Savremeni kompresorski sistemi koji se danas najčešće koriste su [2]:

- Jednoosovinski centrifugalni kompresor (*eng. Single Shaft Centrifugal Compressor*) koji je prikazan na slici 1,
- Centrifugalni kompresor za cevovod (*eng. Pipeline Centrifugal Compressor*),
- Hermetički zatvoren centrifugalni kompresor (*eng. Hermetically Sealed Centrifugal Compressor*),
- Višeosovinski centrifugalni kompresor (*eng. Integrally Geared Centrifugal Compressor*),
- Recipročni ili naizmenični kompresor (*eng. Reciprocating Compressor*).

Zbog kompleksnosti savremenih kompresorskih sistema, sistem automatizacije zahteva određenu arhitekturu kontrolnih sistema tj. ormara upravljanja. Radi lakše kontrole i nadzora celog postrojenja najčešće se arhitektura sistema realizuje na taj način da svaki podsistem postrojenja ima poseban orman upravljanja. S tim u vezi u jednom takvom sistemu postoji orman za upravljanje glavnim motorom kompresora MV-MCC (*eng. Medium Voltage – Motor Control Center*). Upravljanje sistemima za podmazivanje, hlađenje kao i za zagrevanje je uglavnom integrisano u jedan orman LV-MCC (*eng. Low Voltage – Motor Control Center*). Sistem prinudnog zaustavljanja je integrisan u orman ESD (*eng. Emergency Shutdown*), a merno regulacioni sistem u orman UCP (*eng. Unit Control Panel*). Glavni orman u postrojenju je distribuirani kontrolni sistem DCS (*eng. Distributed Control System*), sa kojim se vrši nadzor i



Slika 1. Jednoosovinski centrifugalni kompresor sa horizontalni tipom kućišta [2]

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Boris Dumnić, vanr. prof.

upravljanje svih ormana koji izvršavaju određene kontrolne petlje. Takoreći, DCS predstavlja mozak celog sistema neke fabrike. Arhitekturom sistema se takođe definiše broj razvodnih kutija u polju JB (*eng. Junction Box*), koji služe za prikupljanje signala sa mernih instrumenata, i lokalnih kontrolnih panela LCP (*eng. Local Control Panel*), koji omogućavaju izvršne funkcije operateru na licu mesta.

U ovom radu prikazan je način realizacije kontrolnog sistema za kompresor koji se koristi u hemijskoj industriji za transport hlora. Hlor je gas žuto zelene boje koji je na sobnoj temperaturi u gasovitom obliku. Za ovu konkretnu aplikaciju predviđa se upotreba jednoosovinskog turbokompresora sa 3 procesna nivoa (*eng. Multi Stage Single Shaft Turbocompressor Train*), sa horizontalnim tipom kućišta. Kompresor pogoni električni asinhroni motor snage 350 kW, nazivnog napona 3,3 kV i nominalne brzine 1780 obrtaja po minuti. Pored samog kompresorskog sistema koji služi za transport gasa, za realizaciju ovog pogona potrebna je i implementacija sledećih upravljačkih jedinica:

- Jedinica sa uljem za podmazivanje,
- Jedinica sa azotom, koji služi za gasno zaptivanje sistema,
- Jedinica za hlađenje sistema vodom,
- Jedinica za upravljanje i nadzor glavnog motora,
- Jedinica za nadzor vibracija i brzine kompresora.

2. UPRAVLJANJE KOMPRESORSKIM SISTEMOM

U cilju upravljanja ovakvim sistemom potreban je kontrolni sistem ili kontrolni ormar, koji služi da prikupi procesne signale, obradi ih i emituje određene upravljačke signale ka sistemu. Kontrolni sistem može da se sastoji iz jednog ili iz više kontrolnih ormana, razvodnih ormana u polju i lokalnih kontrolnih stanica, sa ili bez HMI (*eng. Human Machine Interface*) upravljačkih panela. Navedene ormane može obezbediti jedan dobavljač ili više njih u zavisnosti od zahteva klijenta. Ovo sve zavisi od vrste i složenosti aplikacije, na osnovu koje se definiše projektni zadatak.

2.1 Opis projektnog zadatka za realizaciju merno regulacionog sistema

Projektni zadatak se formira na osnovu zahteva klijenta, ali i tehničkih preporuki koje su pripremljene i predstavljene klijentu u fazi prodaje. U ovom delu rada razmatra se projektni zadatak za dizajniranje i proizvodnju UCP-a tj. kontrolnog panela za merno regulacioni sistem.

Pred inženjera su postavljeni sledeći zahtevi vezani za sam kompresorski sistem i prateću opremu:

- SIMATIC S7 PLC sistem,
- SIMATIC TP1200 Comfort Touch Panel HMI,
- Sprežni releji za povezivanje digitalnih signala između UCP-a i ostalih sistema,

- Sekcija u ormanu za prikupljanje signala (*eng. Marshaling*),
- Galvanski izolatori za povezivanja između UCP-a i DCS-a,
- Vibrocontrol VC6000 Compact Monitor, uređaj za nadzor vibracija i brzine kompresora,
- Lampe i tasteri na vratima UCP-a,
- SIMATIC ET 200M – periferni ulazno izlazni modul (*eng. RIO – Remote IO*),
- Razvodne kutije u polju,
- Izvođenje radova i povezivanje instrumentacije.

Na osnovu predviđene pozicije kontrolnog ormana u postrojenju i klasifikacije oblasti u kojoj će orman biti postavljen definisano je sledeće:

- Slobodnostojeći orman u bezbednoj zoni (*eng. non-hazardous area*),
- Bez posebne zaštite od atmosferskog pražnjenja i zemljotresa,
- Temperatura okoline: od +5 do +40°C,
- IP54 tip zaštite,
- Materijal: čelik,
- Standardna debljina zidova ormana,
- Proizvođač: Rittal,
- Tip: VX25,
- Boja ormana: RAL 7035,
- Boja cikle, RAL 9005,
- Bez sistema za grejanje ormana.

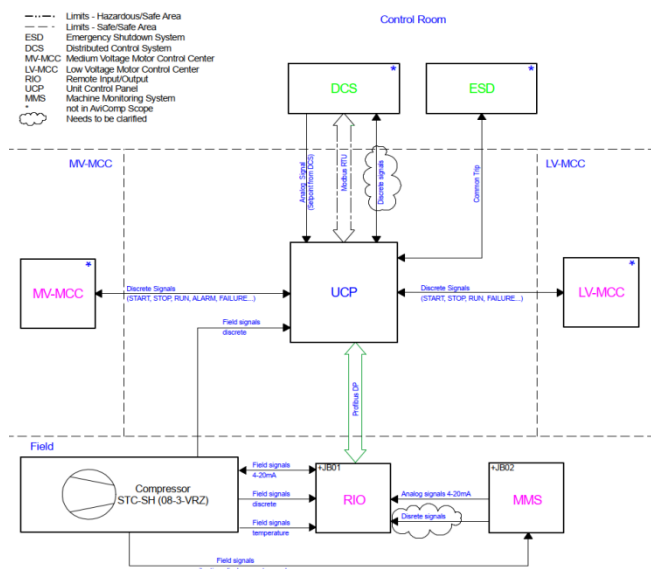
Takođe je potrebno da se obezbedi ventilacija prema proračunu termičke disipacije elemenata u ormaru.

2.2 Realizacija projektnog zadatka – osnovni inženjering

Prvi korak u realizaciji projektong zadatka je izrada osnovnog paketa dokumenata, koji podrazumeva osnovni inženjering. Ulazni podaci se dobijaju iz dva dokumenta:

- Tehnička specifikacija ili projektni zadatak, koji je opisan u prethodnom delu i
 - Procesni dijagram P&ID (*eng. Process & Instrumentation Diagram*).
- Osnovni inženjering obuhvata radni paket, koji se sastoji od sledećih dokumenata:
- Lista signala,
 - Arhitektura sistema, koja je prikazana na slici 2,
 - Tipična kontrolna kola,
 - Lista kablova,
 - Izgled glavnog ormana upravljanja (spoljašnji i unutrašnji),

- Proračun potrošnje snage elemenata u ormaru po napajanjima,
- Proračun disipacije toplote,
- Šema distribucije snage i šema uzemljenja,
- Lista i izgled nalepnica na ormanu,
- Izgled razvodnih kutija u polju (spoljašnji i unutrašnji).



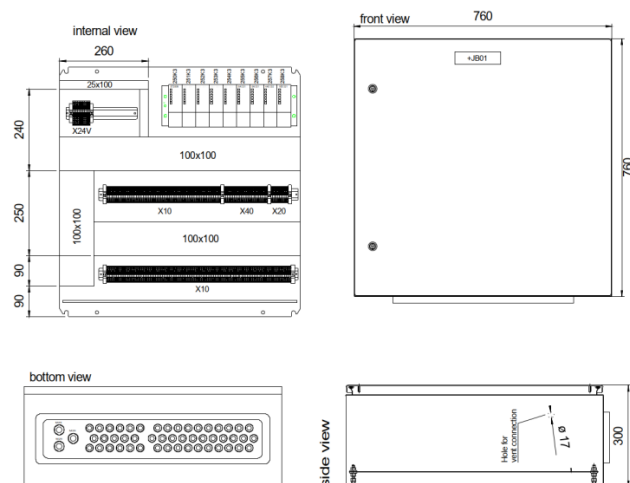
Slika 2. Arhitektura sistema

2.3 Realizacija projektnog zadatka – detaljni inženjering

Paket dokumenata koji podrazumeva detaljni inženjering je sledeći:

- Šema ormara i razvodnih kutija,
- Lista klemu,
- Lista kablova,
- Lista opreme (zasebno po ormarima i razvodnim kutijama).

Pored ovih dokumenata potrebno je kreirati i finalni izgled ormara upravljanja i svih razvodnih kutija. Izgled jedne razvodne kutije prikazan je na slici 3.



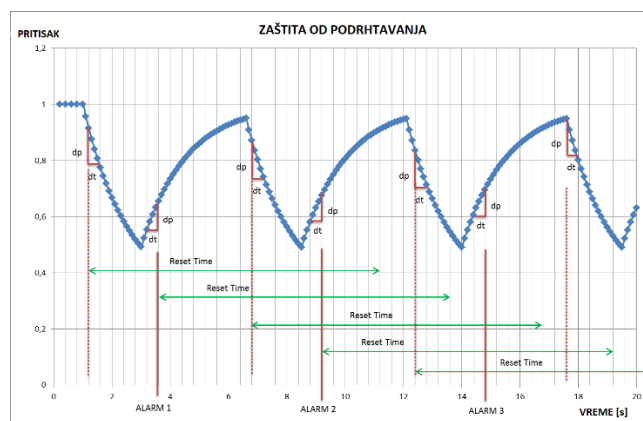
Slika 3. Izgled jedne razvodne kutije

2.4 Funkcionalni opis kompresorskog sistema

Funkcionalni opis kompresorskog sistema je dokument koji klijentu obezbeđuje jasan uvid u logiku softvera i sekvence upravljanja postrojenjem. Glavne funkcije upravljanja postrojenjem koje su integrisane u UCP su sledeće:

- Kontrola performansi (*eng. performance control*),
- Kontrola podrhtavanja kompresora (*eng. anti-surge control*),
- Kontrola pomoćne uljne pumpe (*eng. control of auxiliary oil pump*),
- Kontrola grejača za ulje (*eng. control of oil heater*),
- Kontrola ulazne vodilice (*eng. control of inlet guide vane - IGV*),
- Kontrola šant ventila (*eng. control of bypass valve - BPV*),
- Kontrola ventila za otpadni gas (*eng. control of waste gas valve - WGV*),
- Kontrola ventila za zaptivni gas (*eng. control of seal gas valve - SGV*).

Kontrola BPV ventila je vrlo bitna u pogonima kompresora, jer taj ventil predstavlja glavni kontrolni element takozvane „anti-surge“ kontrole, ili kontrole podrhtavanja kompresora [3]. Podrhtavanje podrazumeva fluktuacije pritiska i protoka u kompresoru, koje mogu dovesti do ozbiljnih mehaničkih oštećenja, pa čak i do uništenja kompresora [4]. Do podrhtavanja uglavnom dolazi kada gas na izlaznoj strani kompresora, iz nekog razloga, ne može da nastavi svoj tok dalje u proces u datom trenutku. U ovakvim situacijama gas se pomoću BPV ventila i cevovoda vraća na ulaz kompresora, čime mu se obezbeđuje recirkulacija ili se pomoću WGV ventila ispušta u atmosferu. Funkcionalnost softvera treba da obezbedi alarmno stanje i stanje isključenja prilikom pojave čestih podrhtaja. Prilikom svakog podrhtaja postepeno se otvara BPV, a kada se desi definisani maksimalni broj podrhtaja u definisanom vremenskom opsegu, otvara se BPV na maksimum, otvara se WGV na maksimum, i generiše se sigurnosni trip koji gasi mašinu. Varijacija pritiska gasa prilikom pojave podrhtavanja kompresora prikazana je na slici 4.



Slika 4. Varijacija pritiska gasa sa uključenom zaštitom od podrhtavanja kompresora

3. PRAVILA RADA SA OPREMOM KOJA SE INSTALIRA U EKSPLOZIVNIM ZONAMA

Kao što je već navedeno kompresorski sistemi velikih snaga se instaliraju za namenu transporta industrijskih gasova. Većina industrijskih gasova su zapaljive prirode, kao što je, na primer, tečni naftni gas a mnogi zapaljivi materijali gore u spoju sa hlorom (npr. sa kiseonikom). Dodatno zapaljivi gasovi i isparenja zapaljivih tečnosti prave sa hlorom eksplozivne smese, hlor reaguje eksplozivno sa mnogo hemijskih supstanci a posebno sa acetilenom, eterom, ugljovodonicima i dr.

Zbog same izloženosti kompletnog postrojenja ovakvim zapaljivim gasovima, vrlo je bitno biti upoznat sa pravilima projektovanja pogona koji se nalaze u eksplozivnim sredinama. Eksplozija predstavlja burnu i destruktivnu reakciju između zapaljive materije, kiseonika i izvora zapaljenja u nekontrolisanim uslovima, slika 5 [5].



Slika 5. Uslovi za nastanak eksplozije [5]

Zarad zaštite ljudskih života, na prvom mestu, i zarad sprečavanja materijalne štete, uvedeni su određeni principi i standardi, koji definišu adekvatne načine zaštite od eksplozija.

3.1 Sekundarna zaštita od eksplozija i principi zaštite

Sekundarna zaštita od eksplozija podrazumeva klasifikaciju opasnih sredina u zone, prema učestanosti javljanja eksplozivne sredine. Klasifikacija opasne sredine u zone podrazumeva definisanje realnih prostornih okvira u kojim se može javiti eksplozivna atmosfera. Za opasne gasove i pare, zone se dele na:

- Zona 0 – definiše prostor u kom se zapaljivi gas ili para javlja vrlo često, ili su kontinualno prisutni.
- Zona 1 – definiše prostor u kom se isti takav gas (para) uglavnom javlja pri normalnom radu.
- Zona 2 – definiše prostor u kom se zapaljivi gas (para) ne pojavljuju u normalnom radu, ili ako se pojave, traju kratak vremenski period. Ovakvu podelu eksplozivnih sredina prihvatili su uglavnom sva tela za standardizaciju, kao što su IEC 60079 standard, koji se poštuje u celom svetu i NEC 505 standard koji se poštuje u Sjedinjenim Američkim Državama.

Principi zaštite su definisani kako bi se ograničio izbor opreme, tj. kako bi se isključila iz upotrebe oprema koja potencijalno može da izazove eksploziju. Za raznovrsnu opremu električnu i neelektričnu definisan je sigurnosni nivo opreme EPL (*eng. Equipment Protection Level*). EPL se deli na EPL a, b i c sa veoma visokim, visokim i normalnim nivoom zaštite, respektivno. Postoje četiri

principa zaštite koji mogu sprečiti da oprema postane izvor zapaljenja. U zavisnosti od implementacije opreme, primenjuje se i odgovarajući princip zaštite. Ono što je cilj sva četiri principa je da obezbedi, da oprema ne dostigne temperaturu zapaljena okolne sredine. Svi načini za zaštitu od eksplozija i za definisanje principa zaštite su predstavljeni u standardima EN IEC 60079-0 za električne uređaje i EN ISO 80079-36, -37 za ne električne uređaje [5].

4. ZAKLJUČAK

Savremena industrija nafte i gasa, kao i različite grane hemijske industrije zahtevaju upotrebu kompresorskih sistema velikih snaga u cilju transporta industrijskih gasova. Upravljanje kompresorskim sistemima predstavlja zahtevan inženjerski zadatak koji zahteva realizaciju kompleksnih upravljačkih sklopova kao i programa za upravljanje. Dodatni izazov predstavlja i činjenica da transport industrijskih gasova najčešće podrazumeva i rad sa opremom koja se instalira u eksplozivnim zonama. U ovom radu su prikazani neki od osnovnih inženjerskih zadataka koji se očekuju prilikom realizacije merno regulacionog sistema jednog savremenog kompresorskog sistema velike snage.

5. LITERATURA

- [1] S. Mokhtab, W. A. Poe and J. G. Speight „Handbook of Natural Gas Transmission and Processing“, Elsevier, 2006.
- [2]<https://new.siemens.com/global/en/products/energy/compression.html>
- [3] T. Besselmann, A. Cortinovis, and all „Control and Productivity, Saving the day – electric driven gas compressor control“ ABB review, July 2017.
- [4] A. Cortinovis, D. Pareschi, M. Mercangoez, T. Besselmann „Model Predictive Anti-Surge Control of Centrifugal Compressors with Variable-Speed Drives“, 2012 IFAC Workshop on Automatic Control in Offshore Oil and Gas Production
- [5] „Basic concepts for explosion protection“, dostupno na <https://www.bartec.de/en/safety-academy/basic-concepts-for-explosion-protection/>

Kratka biografija:



Luka Janićić rođen je u Rumi 1994. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine odbranio je 2017. god. Master rad na Fakultetu tehničkih nauka iz oblasti Energetska elektronika i električne mašine odbranio je 2019.god.

kontakt: janicicluka@gmail.com



Boris Dumnić je diplomirao 2003. godine na Univerzitetu u Novom Sadu, Fakultetu tehničkih nauka iz oblasti elektrotehnika i računarstvo. Od 2004. godine zaposlen je na Fakultetu tehničkih nauka u Novom Sadu. Magistarski rad je odbranio 2007. godine a 2013 godine je odbranio i doktorsku tezu. Oblast interesovanja su mu elektromotorni pogoni, energetska elektronika i obnovljivi izvori električne energije.

KRATKOROČNA PROGNOZA REŽIMA DISTRIBUTIVNE MREŽE**SHORT TERM FORECASTING OF THE DISTRIBUTION NETWORK STATE**Nikola Jerković, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – Rad treba da proveriti tačnost rezultat algoritma teorije vektora podrške koji se koristi za proračun prognoze opterećenja izvoda distributivne mreže i pokaže da se tako dobijeni rezultati koriste za kalibraciju potrošnje koja pripada tom izvodu i pokrene proračun tokova snaga i indeksa performansi u cilju otkrivanja najkritičnijih elemenata u mreži za svaki prognozirani trenutak.

Gljučne reči: proračun prognoze, mašinsko učenje, teorija vektora podrške.

Abstract – The work needs to verify the accuracy of the result of the support vector machine algorithm used to calculate the load forecast of the distribution network and show that the results thus obtained are used to calibrate the consumption of that statement and trigger the load flow and performance index calculation in order to detect the most critical elements in the network for each forecasted moment.

Gljučne reči: load forecast, machine learning, support vector machine.

1. UVOD

Privredni razvoj modernih zemalja usko je povezan sa dostupnošću električne energije. Pored uglja, drveta, nafte, prirodnog gasa i drugih energenata koji su kroz istoriju bili dominantni u industriji, električna energija kao najpogodniji vid energije za prenos i transformaciju u druge oblike zaslužno zauzima sve veći procenat upotrebe u industriji. Pored industrijskih potrošača električne energije značajnu ulogu čine i domaćinstva čija je potrošnja iznosila 51% od ukupne potrošene energije u republici Srbiji 2017. godine [1]. Za njenu kontinualnu, pouzdanu i kvalitetnu isporuku zadužena su distributivna preduzeća koja se svakodnevno susreću sa izazovima kao što su održavanje mreže, otklanjanje kvarova, upravljanje opterećenjem, kupovina i prodaja električne energije. U modernim distributivnim preduzećima sve više se koriste softverska rešenja koja za cilj imaju da olakšaju posao dispečerima u svakodnevnom radu i uz to povećaju prihode i smanje rashode distributivnih preduzeća.

Danas je prognoza potrošnje električne energije važan proces u elektroenergetskim preduzećima sa primenom u nekoliko sektora, kao što su sektor za planiranje, operativu, prodaje itd. Potrebe za kvalitetnom prognozom potrošnje električne energije vezane su za sledeće aktivnosti preduzeća [2]:

NAPOMENA:

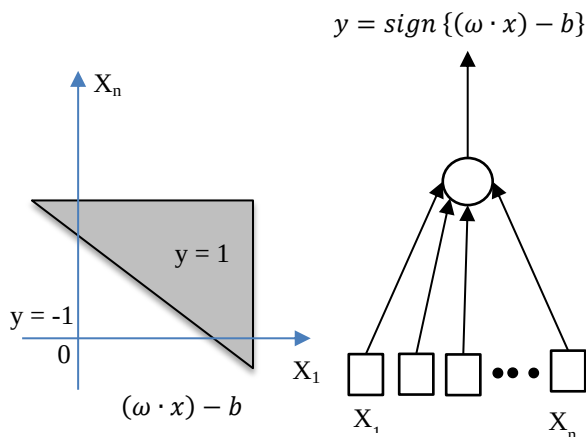
Ovaj rad proistekao je iz master rada čiji je mentor bio dr Goran Švenda, red. prof.

1. Kupovina energije. Prognoza potrošnje električne energije je osnova za proces kupovine energije, bez obzira da li taj proces obavlja samo preduzeće ili taj proces prepušta nekoj spoljnoj, nezavisnoj organizaciji. Preduzeće može da obavlja bilateralnu kupovinu sa obavezom na duži period od npr. 10 godina, može da rezerviše i kupuje na intervalima od jednog meseca do 3 godine unapred ili da prilagođava i kupovinu obavlja na tržištu za dan unapred.
2. Planiranje prenosa i distribucije. Preduzeća imaju potrebu za održavanjem i blagovremenim unapređenjem sistema kako bi se zadovoljile potrebe rasta potrošnje, obezbedilo stabilno snabdevanje i povećala pouzdanost sistema. Preduzeća moraju da planiraju i kupovinu zemljišta u budućnosti kako bi se stvorila mogućnost za izgradnju novih elektroenergetskih objekata. Planiranje prenosa i distribucije se u velikoj meri oslanja na prostornu prognozu potrošnje koja treba da pruži odgovore na pitanja koliko i gde će porasti potražnja za električnom energijom.
3. Održavanje i operativa. Na dnevnom nivou, dijagrami opterećenja dobijeni prognozom potrošnje, omogućuju dispečeru da donosi pravovremene odluke o manipulaciji i planira održavanje.
4. Upravljanje opterećenjem (eng. Demand side management – DSM). Iako većina aktivnosti DSM-a spada u dnevne aktivnosti, poželjno je odvojiti DSM kao poseban proces zbog njegovog značaja kod "smart grid" koncepta. Prognoza potrošnje električne energije pomaže donošenju pravih odluka kad su u pitanju regulacija opterećenja i smanjivanje napona. Takođe, pomaže da preduzeća sprovedu dugoročno planiranje bazirano na dobijenim karakteristikama opterećenja krajnjih korisnika tokom procesa prognoze [3].

2. OSNOVNI POJMOVI

Teorija vektora podrške (*Support Vector Machine – SVM*) je inicijalno razvijan sa ciljem da se unaprede neuronske mreže. Ona je bazirana na principu strukturnog smanjenja rizika (*Structural Risk Minimisation – SRM*), koji je u velikoj meri bolji od empirijskog smanjenja rizika (*Empirical Risk Minimisation – ERM*), koji se koristi kod neuronskih mreža. SRM minimizuje gornju granicu rizika, dok ERM minimizuje donju granicu rizika i grešku u trening podacima [4]. Osnovni princip rada ove metode je da se obučavanje vrši na osnovu datih podataka u vidu obučavajućeg skupa, koji se sastoji od skupa ulaznih i skupa izlaznih podataka. Zatim se, ovako obučena struktura, koristi za rešavanje problema klasifikacije i regresije. Ovu metodu je, sa saradnicima, početkom

devedesetih godina prošlog veka razvio Vapnik (*Vladimir Naumovich Vapnik*) [5]. Teorija mašinskog učenja svoj razvoj započinje pojavom koncepta perceptrona, koji je predložio Rozenblat (*Frank Rosenblatt*) [6]. Njegov model rešava problem prepoznavanja, odnosno razdvajanja podataka u dve kategorije. Predloženi perceptron je za osnovu uzimao već razvijeni model veštačkog neurona. Razbijeni neuron je imao n ulaza i jedan izlaz iz skupa $\{-1,1\}$, koji kao takav deli prostor ulaznih podataka na dva dela, Slika 1.



Slika 1. Perceptron

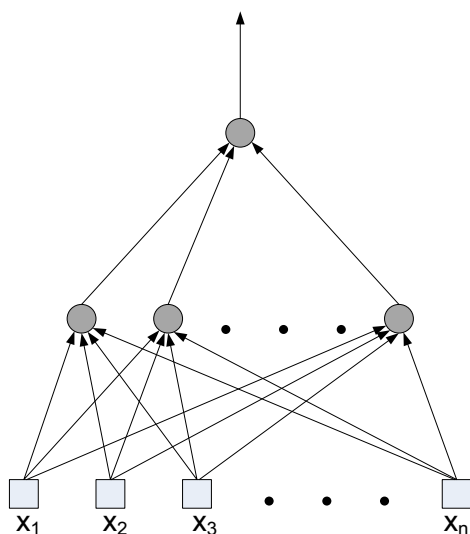
Funkcija zavisnosti izlaza od ulaza data je sa [4]:

$$y = \text{sign} \{(\omega \cdot x) - b\}, \quad (1)$$

gde je x vektor ulaznih podataka, b prag osetljivosti. Hiperravan koja razdvaja dva skupa data je sa:

$$(\omega \cdot x) - b = 0. \quad (2)$$

Model koji je napravio Rozenblat se sastojao od više neurona, raspoređenih po nivoima, tako da su izlazi jednog nivoa bili ulazi u sledeći nivo, Slika 2, dok se u poslednjem nivou nalazi samo jedan neuron čiji izlaz predstavlja izlaz cele mreže.



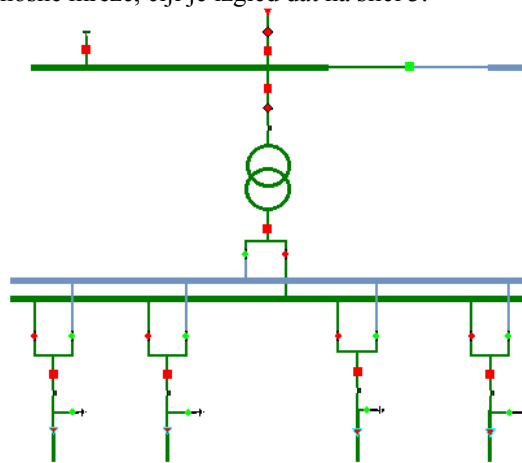
Slika 2. Model sa više neurona raspoređenih po nivoima

Gledano geometrijski, perceptron deli prostor x na dva dela, granicom razdvajanja koja može, ali i ne mora biti linearna. Podela se postiže obučavanjem mreže, što se u

stvari svodi na odabir pravih koeficijenata za neurone u mreži. Kako se javilo simultano menjanje koeficijenta neurona u perceptronu, predloženo je da se fiksiraju koeficijenti svih neurona osim poslednjeg, gde je cilj bio da se tokom procesa obuke pronađu koeficijenti poslednjeg neurona [6]. Ovim se prostor ulaznih podataka x transformiše u novi prostor z , u kojem se, na osnovu podatak iz obučavajućeg, konstruiše odgovarajuća hiperravan. Vektor ω se inicijalizuje na 0, a zatim ostaje isti ili se menja po određenom pravilu, u zavisnosti od toga da li je ω element obučavajućeg skupa dobro klasifikovan ili ne.

3. MATEMATIČKI MODEL

Za potrebe proračuna koristi se model mreže koji se sastoji od četiri izvoda, njihovog napojnog transformatora 110 / 20 kV snage 20 MVA i korena koji predstavlja kraj prenosne mreže, čiji je izgled dat na slici 3.



Slika 3. Test mreža

Pre početka proračuna potrebno je izabrati elemente mreže za koje će biti izvršen proračun, oni mogu biti: izvodi, transformatori, transformatorske stanice kao i celokupna mreža. Pored izbora elemenata potrebno je izabrati dužinu vremenskog intervala za proračun koji može biti između jednog i 24 sata, sa korakom od 15, 30 ili 60 minuta. Proračun počinje tako što se za sve izabrane elemente odrede njihovi izvodi i za svaki pokrene proračun vektora podrške, za izabrani vremenski interval u budućnosti. koji je objašnjen u prethodnom poglavlju. Ovaj proračun se koristi se za prognozu aktivne snage. Parametri koji se uzimaju kao ulazni za algoritam teorije vektora podrške su:

- Istorijski podaci o potrošnji za tri godine unazad.
- Istorijski podaci o vremenu za tri godine unazad i to za temperaturu, brzinu vetra i oblačnost.
- Tipove dana za tri godine unazad i dva dana unapred – razlikuje se: radni dani, dani vikenda i praznični dani.
- Istorijski podaci o danim obeleženim kao neželjeni za tri godine unazad. Ovde se podrazumevaju dani koje se iz nekog razloga žele isključiti iz skupa za trening, npr. operater je svestan da tog dana merenja protoka nisu bila validna za trening iz nekog razloga.

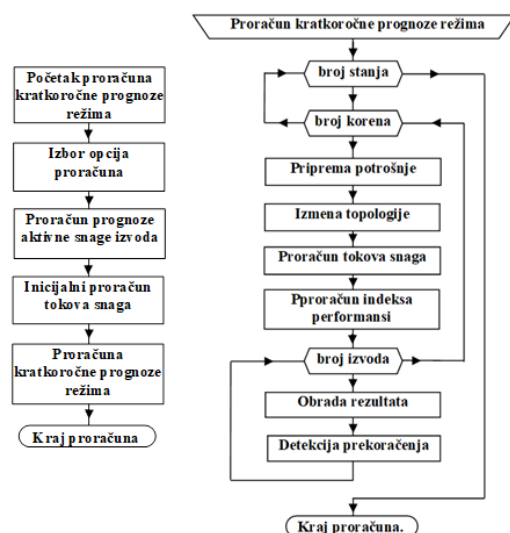
Rezultat ovoga proračuna sačinjavaju nizovi satnih vrednosti aktivne snage potrošnje po izvodu koji su spremni za dalju obradu.

Nakon završenog proračuna prognoze, rezultati aktivne snage izvoda se kalibracijom raspoređuju na njegove potrošače vodeći računa o tome koliki je izabrani korak. Obzirom da su rezultati prognoze aktivne snage izvoda uvek nizovi sa satnim vrednostima, kalibrisane snage potrošača za korake od 15 ili 30 minuta, biće dobijeni linearnom interpolacijom dva susedna satna rezultata. Tako da će u modelu uz svakog potrošača biti smešteni i niz vrednosti koje će opisivati promenu njegove potrošnje tokom izvršavanja proračuna.

Posle završene kalibracije pokreće se inicijalni proračun tokova snaga za svaki koren koji predstavlja ulaz u distributivnu mrežu, gde se za sve potrošače postavlja prva vrednost iz niza koji opisuje ponašanje potrošnje, u cilju provere konvergencije proračuna. Koreni za koje tokovi snaga ne konvergiraju neće se dalje razmatrati. Blok prikazan na Slici 4 (levo), predstavlja prethodno opisane etape pripreme za proračun kratkoročne prognoze režima.

Sam proračun kratkoročne prognoze režima počinje nakon završene pripreme i sastoji se od tri ugnježene petlje koje omogućavaju da se prođe kroz sve vremenske intervale, a unutar svakog kroz sve korene za koje se izvršava proračun tokova snaga i indeksa performansi. Poslednja petlja prolazi kroz sve elemente za čiji je koren završen proračun tokova snaga i indeksa performansi i filtrira željene rezultate koji će biti izlaz iz proračuna.

Blok koji je zadužen za pripremu potrošnje treba za svaki potrošač u mreži postaviti potrošnju aktivne snage na vrednost iz prethodno iskalibrisanog niza na onu koja odgovara vremenskom trenutku za koji se izvršava proračun. Što je se ponavlja u svakoj iteraciji kroz stanja. U koliko u vremenskom trenutku za koji se vrši proračun prognoze postoji planirana prekiđaćka manipulacija njen efekat će se uvažiti promenom topologije mreže. Nakon pripremljene potrošnje i izmene topologije pokreće se proračun tokova snaga za korene, čime se dobija režim mreže za datu potrošnju i topologiju.



Slika 4. Algoritam proračuna kratkoročne prognoze režima mreže

Proračun indeksa performansi se oslanja na proračun tokova snaga i pokreće se nakon njega. Dobijeni rezultati daju detaljnije podatke vezane za elemente mreže kao što su: struja, napon, aktivna snaga, reaktivna snaga u relativnim i apsolutnim jedinicama. Pored toga za sve

elemente detektuju se prekoračenja limita napona, struje, aktivne snage, disbalansa opterećenja i obrnutog toka snage. Kada je izvršen proračun indeksa performansi, popunjavaju se podaci potrošnje, injektirane i generisane aktivne i reaktivne snage. Ukoliko postoje problemi sa prekoračenjima limita za bilo koji element u mreži, ta prekoračenja se alarmiraju za trenutak u kome će se desiti. Blok dijagram algoritma prikazan je na slici 4.

4. VERIFIKACIJA REZULTATA

Verifikacija proračuna kratkoročne prognoze režima mreže je izvršena na četiri izvoda i za svaki je proračun pokrenut za četiri dana od kojih je svaki iz jedne godišnje sezone. Pokretanje proračuna je rađeno tako što je lokalno vreme na mašini vraćano u nazad na razmatrane karakteristične dane. Na taj način bilo je jednostavno uporediti prognoziranje vrednosti i vrednosti iz istorije, odnosno utvrditi koliko je algoritam za prognoziranje tačan. Tačnost samog algoritma analizirana je na osnovu apsolutne srednje procentualne greške (*Mean Absolute Percentage Error – MAPE*) [7]:

$$MAPE = \frac{100}{n} \sum_{t=1}^n \left| \frac{A_t - F_t}{A_t} \right|, \quad (4.1)$$

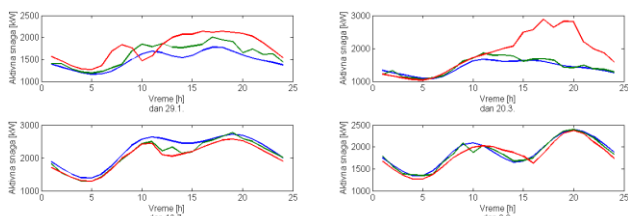
gde je sa n naznačen broj sati za koje se MAPE računa. Sa A_t su naznačene stvarne vrednosti, podaci iz istorijske baze, a sa F_t na osnovu algoritma izračunate vrednosti prognoze. Razlog za korišćenje ove tehnike je vrlo jednostavan, ljudima je lako da razumeju procenat, čak i ako ne razumeju u potpunosti materiju koju procenti izražavaju, odnosno jednostavno mogu da ustanove kolika je greška proračuna. Što je veća vrednost za MAPE, to je veća greška proračuna. Tabele 1 predstavlja rezultate koji za ulaz koriste neposrednu istoriju dok u Tabeli 2 ulaz u proračun je dobijen algoritmom pretraživanja sličnih dana. Vrednosti prikazane u tabelama predstavljaju srednje apsolutne procentualne greške i maksimalne procentualne greške za četiri izvoda jedne transformatorske stanice za četiri karakteristična dana od kojih je svaki iz jedne godišnje sezone, dok su na slikama 5, 6, 7 i 8 grafički prikazani rezultati dva prethodno navedena proračuna. Plava linija predstavlja istorijske podatke sa kojima se vrši poređenje rezultata, crvenom linijom su predstavljeni rezultati proračuna dobijeni sa skorijom istorijom, a zelenom podaci dobijeni algoritmom sličnih dana.

Tabela 1. Vrednosti MAPE i maksimalne greške za period od 24 sata, rezultati dobijeni na osnovu bliske istorije

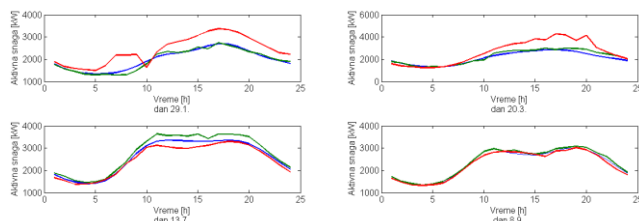
	29.1.		20.3.		13.7.		8.9.	
izvod	MAPE ₂₄	Najveća greška [%]	MAPE ₂₄	Najveća greška [%]	MAPE ₂₄	Najveća greška [%]	MAPE ₂₄	Najveća greška [%]
1	2,27	10,32	5,17	11,77	5,53	14,66	2,28	10,32
2	3,72	13,67	6,51	15,56	6,28	9,86	3,01	5,79
3	8,15	14,92	8,51	15,61	8,97	14,64	7,35	14,42
4	7,96	14,64	7,19	13,48	3,49	7,87	4,78	7,85

Tabela 2. Vrednosti MAPE i maksimalne greške za period od 24 sata, rezultati dobijeni algoritmom sličnih dana

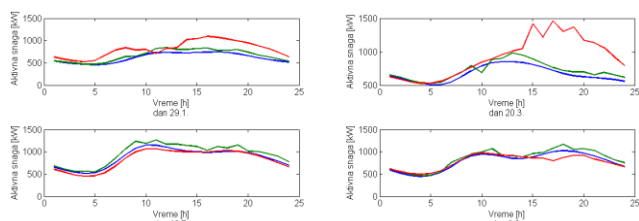
	29.1.		20.3.		13.7.		8.9.	
izvod	MAPE ₂₄	najveća greska [%]	MAPE ₂₄	najveća greska [%]	MAPE ₂₄	najveća greska [%]	MAPE ₂₄	najveća greska [%]
1	20,25	36,72	31,50	59,42	8,01	19,06	5,96	14,06
2	22,28	56,19	23,60	63,99	4,40	10,25	1,87	6,79
3	31,40	59,41	38,40	112,95	5,74	12,76	6,28	20,70
4	41,51	104,20	60,02	174,62	6,99	8,79	9,07	44,05



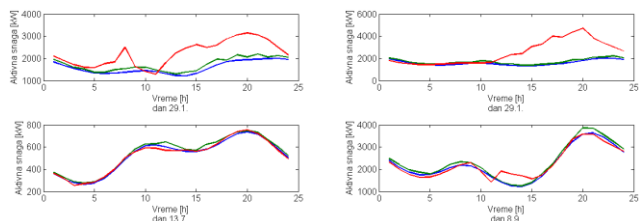
Slika 5. Rezultati za izvod 1 za četiri karakteristična dana



Slika 6. Rezultati za izvod 2 za četiri karakteristična dana



Slika 7. Rezultati za izvod 3 za četiri karakteristična dana



Slika 8. Rezultati za izvod 4 za četiri karakteristična dana

Analizom prikazanih rezultata može se uočiti da se primenom algoritma sličnih dana dobijaju znatno bolji rezultati. Primenom tog algoritma prevaziđen je problem pojave pikova prognoziranih veličina sa jako velikim greškama. Oblik dijagrama koji su dobijeni primenom algoritma sličnih dana ne odstupa od oblika istorijskih dijagrama (kao što je slučaj kod rezultata dobijenih algoritmom koji radi sa bliskom istorijom). Konačno, može se uočiti da je primenom algoritma sličnih dana značajno smanjena vrednost MAPE (za neke trenutke sa vrednosti šezdeset vrednosti ispod osam procenta), isto tako smanjena je i maksimalna greška (sa preko sto na ispod šesnaest procenata).

5. ZAKLJUČAK

Prednost korišćenja proračuna tokova snaga i indeksa performansi u kombinaciji sa SVM algoritmom je u tome što omogućava prognozu režima i obezbeđuje sve ključne podatke, kao što su: struja, napon, aktivna snaga, gubici aktivne snage itd. za sve elemente distributivne mreže. Pored svega prethodno navedenog, u okviru proračuna indeksa performansi, postoji i detekcija kritičnih elemenata sa njihovim prekoračenjima limita koji su dati u apsolutnim i relativnim vrednostima. Proračuni tokova snaga i indeksa performansi za svaki prognozirani vremenski trenutak otvaraju razne mogućnosti vezane za analizu, poboljšanje

performansi i angažovanja distribuiranih elektroenergetskih izvora mreže. Prednost ovog proračuna je ta što se uvažavaju i izmene topologije mreže određene planiranim prekidnim manipulacijama i time se dodatno povećava uvid u ponašanje mreže. Proračun može da se proširi sa algoritmom koji određuje da li će određeni distribuirani elektroenergetski resurs moći da radi sa unapred definisanom krivom potrošnje, odnosno proizvodnje i kao rezultat vraćati krivu po kojoj treba da radi kako ne bi dolazilo do problema u mreži. Potreba za poboljšanjem proračuna svakako postoji, iz razloga što postoje odstupanja u oblicima krivih koje se porede.

Neki od pravaca daljeg unapređenja algoritma su:

- Realizovati proračun u kratkim vremenskim intervalima, čime bi se češće osvežavali rezultati, odnosno smanjila greška procene.
- Omogućiti da se za različite sezone i oblike krive potrošnje primenjuju različiti algoritmi, npr. različite varijante SVM-a i neuronske mreže (za različite konfiguracije mreža, broja neurona, dubine mreže i metoda za obuku).
- Kao ulazni parametar proračuna prognoze koristiti i topologiju mreže, za koju je ostvarena potrošnja.
- Omogućiti da se prognoza radi istovremeno za više od jedne tačke na izvodu.

6. LITERATURA

- [1] Energetski bilans Republike Srbije za 2017. god.
- [2] T.Hong: *Short Term Electric Load Forecasting*, North Carolina State University, 2010.
- [3] M.M.Božić: *Kratkoročna prognoza potrošnje električne energije zasnovana na metodama veštačke inteligencije*, Univerzitet u Beogradu, Elektronski fakultet, Niš, 2014.
- [4] N.Turker, F.Gunes: A competitive approach to neural device modeling: Support vector machines, *Artificial Neural Networks – ICANN 2006. ICANN 2006. Lecture Notes in Computer Science*, Vol. 4132. Springer, Berlin, Heidelberg
- [5] B.E.Boser, I.Guyon, V.N.Vapnik: A training algorithm for optimal margin classifiers; *In Computational Learning Theory*, pp. 144-152, Pittsburgh, Pennsylvania, USA, July 27-29, 1992.
- [6] F.Reossenlat: The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain, *Psychological Review*, Vol. 65, No.6, pp. 386-408, 1958.
- [7] S.Kim, H.Kim: A new metric of absolute percentage error for intermittent demand forecasts, *International Journal of Forecasting*, Vol. 32, No. 3, pp. 669-679, 2016.

Kratka biografija:



Nikola Jerković rođen je 26.10.1992. u Somboru. Srednju tehničku školu u Somboru završava 2011. iste godine upisuje Fakultet tehničkih nauka na kojem stiče zvanje diplomirani inženjer elektrotehnike i računarstva sa diplomskim radom na temu „Realizacija aplikacije na personalnom računaru za komunikaciju sa mikrokontrolerom“, godine 2016. Master rad brani u julu 2019. Živi i radi u Novom Sadu.

MODELOVANJE MALE HIDROELEKTRANE NA BRANI KOD NOVOG BEČEJA**MODELING SMALL HYDRO POWER PLANT AT THE NOVI BEČEJ DAM**

Andor Rajšli, Vladimir Katić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Osnovni cilj ovog rada je predstavljanje potencijala izgradnje male hidroelektrane (ME) kod Novog Bečaja. Pažnja je usmerena na teorijsku mogućnost, na analizu osnovnih merenja i na poređenje sa drugim rezultatima.

Ključne reči: Mala HE, Obnovljivi izvori, Brana

Abstract – The main goal of this paper is to present the potential for the construction of a small hydroelectric power plant near Novi Bečej. Attention is focused on the theoretical possibilities, on the basis of the measurement and finally, the comparison with other results.

Keywords: Small hydropower, Renewable sources, Dam

1. UVOD

Ovaj rad će dati uvid u pretvaranje energije vode u električnu energiju. Da bi se potencijal energije vode mogao iskoristiti potrebne su hidroelektrane. Zahvaljujući geografskim osobinama, Srbija ima dobre mogućnosti za izgradnju hidroelektrana u državi. Ukupna prosečna snaga svih srpskih vodotoka je 31.000 GWh, i većina potencijala je iskorišćena pomoću većih kapaciteta generatora, a ostali su iskorišćeni pomoću više malih, ali skupljih generatora. Za izgradnju malih proizvodnih prostora u Južnoj Srbiji najpovoljnija mesta su Užički, Niški i Kragujevački region, a u Vojvodini to su mesta pored reka Dunava i Tise. U Srbiji je realizovano 31 mini hidroelektrana ukupne snage 34.5 MW [1].

Prednosti malih hidroelektrana su u tome da su veoma pouzdane i elastične u radu. Imaju veoma dug period eksploatacije, građevinski objekti izdržavaju do 100 godina, uz minimalno održavanje. Električne komponente koji se ugrađuju u male HE se retko kvare, i lako se zamenjuju. Mali hidro sistemi mogu da rade bez prisustva ljudi i kontrola može da se vrši daljinskim komandama. Mane malih HE leže u tome što imaju značajan uticaj na ekologiju mesta gde se grade. Sa branom imaju dodatni uticaj na ribnu populaciju, može prezasiti vodu gasovima iz vazduha, čak uticati i na njihovo uginuće [8]. Sredinom Vojvodine teče najvažnija reka regiona, reka Tisa. Na levoj strani obale reke se nalazi grad Novi Bečej kome pripada brana, koji je izgrađena 1977. godine. Brana ima sedam prelivnih polja i brodsku prevodnicu za brodove do 1000 tona nosivosti. Branom se pregrađuje korito Tise za veliku vodu u dužini od 520m. Na stubovima brane projektovan je drumski most, kao saobraćajna veza između Banata i Bačke [2].

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji je mentor bio dr Vladimir Katić, red.prof.

Ovaj rad ima za cilj da prezentuje i objasni potencijal proizvodnje električne energije na Tisi, kod Novog Bečaja. Pored prikaza rezultata merenja o vodostaju, protoku vode i proračuna snaga elektrane, napisan je i idealni izbor turbine kojom je vršeno pretvaranje energije vode u električnu energiju.

2. OPŠTE O HIDROELEKTRANAMA

Da bi se izvršila podela hidroelektrana (HE), bitne su sledeće karakteristike: snaga, visina pada vode i način korišćenja vode. Među hidroelektranama nema saglasno definisane oblasti za velike i male elektrane, ali granicase razlikuje u raznim državama: u Italiji iznosi 3 MW, u Francuskoj 8MW, u Britaniji 5 MW, u SAD 1÷30 MW, u Kini 0.5÷25 MW, a u Indiji 1÷15 MW. Na osnovu visine pada brane se dele u tri kategorije: sa niskim, sa srednjim i sa velikim padom [3]. Za niski pad tipični primeri su u dolinama reka, u grupi sa srednjim padom (30-100 metara) često se grade brane u kombinaciji sa rezervoarom i kanalom kojim se voda dovodi do turbine. Elektrane sa velikim padom grade se uglavnom u brdskim predelima. Prema načinu korišćenja vode razlikuju se: protočne, akumulacione i reverzibilne HE. Protočne hidroelektrane se stavljaju na isti nivo kao nivo na kojem voda teče. Kinetička energija vode direktno se koristi da se pokrene turbina. Akumulacijske HE koriste potencijalnu energiju akumulacionog jezera, a reverzibilne HE predstavljaju specifičnu vrstu elektrana, zato što se dobijena voda, koja teče kroz turbine, nakon iskorišćenja, iz donjeg bazena ponovo pumpa u gornji bazen (akumulacono jezero). Poznata elektrana ovakvog tipa u Srbiji je Bajina Bašta [4].

2.1. Osnovi delovi sistema hidroelektrane

Da bi se pretvorila energija vode u električnu potrebno je računati da su neki delovi elektrane u direktnom dodiru sa vodom, odnosno služe za njeno dovodenje, prikupljanje ili odvođenje. Zbog lakše organizacije komponente možemo podeliti u tri grupe: građevinske-, električne i mašinske komponente. Pod građevinske komponente spadaju brana, mašinska zgrada i cevovodi. Brana je hidrotehnička građevina koja pregrađuje vodotok ili drugu vodenu masu radi zadržavanja nivoa vode na željenoj visini. Mašinska zgrada, u širem smislu je skup građevina i opreme koja se koristi za pretvaranje energije vode u električnu energiju i njenu predaju u distribucionu mrežu. Pod električne komponente spada generator, koji služi za pretvaranje mehaničke energije u električnu. Mašinske komponente HE predstavljaju turbina, multiplikatori i ustava.

2.2. Podela turbine prema načinu pretvaranja energije vode u kinetičku energiju

Razlikuju se dve vrste turbine: impulsne i reakcijske.

Impulsne turbine izgrade se uz pomoć cevi, da bi se povećala brzina vode koja se dovodi na lopatice turbine, kako bi ih okretalo.

Lopatice su obično u obliku kašike. Najpoznatije turbine ovog tipa su Peltonova, Turgo i Bánki-Michell turbine.

Reakcijske turbine kod kojih se u radnom kolu menjaju sve tri komponente energije (pritisak, kinetička i potencijalna). Kod ovih turbina pritisak vode je veći na ulazu nego na izlazu, zbog toga je potrebno da turbina bude zatvorena.

Lopatice turbine mogu biti fiksirane ili pokretne.

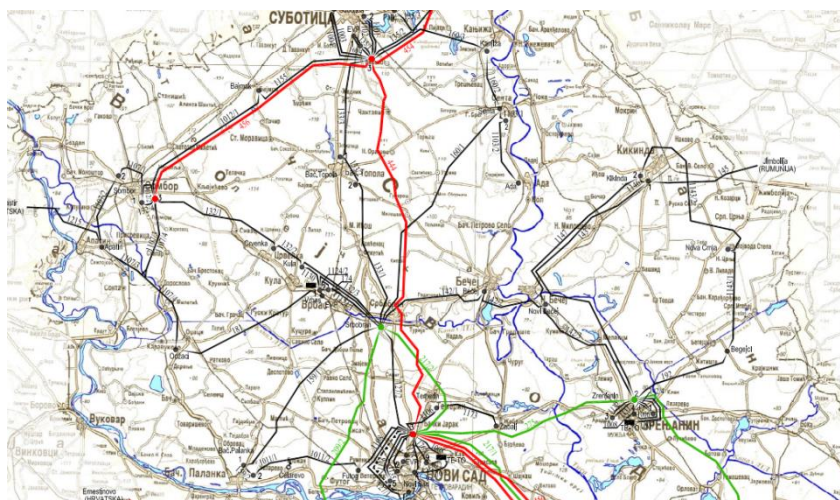
Najčešće korišćene turbine ovog tipa su Francis-ova-, Kaplanova-, Deriaz turbina.

3. ANALIZA MOGUĆNOSTI IZGRADNJE HE NA TISI KOD NOVOG BEČEJA

U Srbiji da bi se izgradila mala hidroelektrana potrebno je da se ispune sledeći uslovi:

- ✓ pribavljanje informacije o lokaciji ili lokacijskim uslovima i izrada tehničke dokumentacije
- ✓ pribavljanje energetske dozvole
- ✓ pribavljanje građevinske dozvole
- ✓ građenje objekta
- ✓ tehnički pregled objekta i pribavljanje upotrebne dozvole

Pre priključenja objekata na elektroenergetsku mrežu obavezno treba da se izvrši i provera kvaliteta električne energije koju elektrana proizvodi, analiza dinamičkih prelaznih procesa, provera usaglašenosti objekata sa Pravilima o radu prenosnog sistema[5]. Priključenje na električnu mrežu omogućeno na 110 kV (slika 1).



Slika 1. Mapa prenosnog sistema oko Novog Bečaja(crvena 400 kV; zelena 220 kV; crna 110 kV) [6]

4. REZULTATI MERENJA

Osnovni podaci i regulacija može da se traži od vlasnika brane, javnog vodoprivrednog preduzeća „Vode Vojvodine“. Karakteristike režima proticaja reke Tise u profilu brane su:

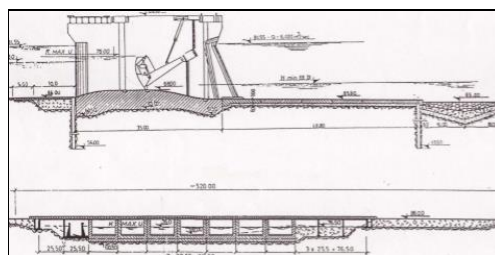
- Minimalni zabeleženi proticaj $Q = 80 \text{ m}^3/\text{s}$
- Srednji proticaj $Q = 708 \text{ m}^3/\text{s}$
- Maksimalni zabeleženi proticaj $Q = 3900 \text{ m}^3/\text{s}$

Podatak o proticanju vode kod Novog Bečaja nije direktno meren. Na teritoriji AP Vojvodine do Novog Bečaja dodavanje količine vode dolazi kod Bečaja i oduzimanje vode se dešava ispred brane. Iz ovoga sledi da količina vode koja se meri u prethodnom gradu od Novog Bečaja potrebno je oduzeti $10 \text{ m}^3/\text{s}$, i ovom aproksimacijom dobija se proticanje kod Novog Bečaja. Visina vode se meri ispred brane (tkz. gornja voda) i direktno posle brane (tkz. donja voda).

Po konstrukciji brane pripada tipuniskog pada. Tipične osobine ovakvih vrsta građevina su u tome da se sastoji od više kapija, koje se mogu regulisati, otvoriti i zatvoriti, čime se reguliše stabilnost reke.

Broj ovakvih kapija u ovom slučaju iznosi sedam takozvanih polja (slika 2).

Da bi se odredila snaga koja može da se proizvede na osnovu potencijala reke Tise, potrebno je dobro znati razliku između gornje i donje vode, i druga najvažnija veličina je proticaj vode kroz branu. Zaključak snage i projektovanje elektrane u ovom radu urađeno je na osnovu analiziranog perioda između 2007-2017.



Slika 2. Podužni i poprečni profil brane [7]

5. PRORAČUN ENERGIJE

Formula za izračunavanje snage vode i snage na turbini su:

$$P_{\text{voda}} = \rho \cdot Q \cdot g \cdot \Delta h \quad (1)$$

$$P_{\text{turbina}} = \eta_{\text{turbina}} \cdot P_{\text{voda}} \quad (2)$$

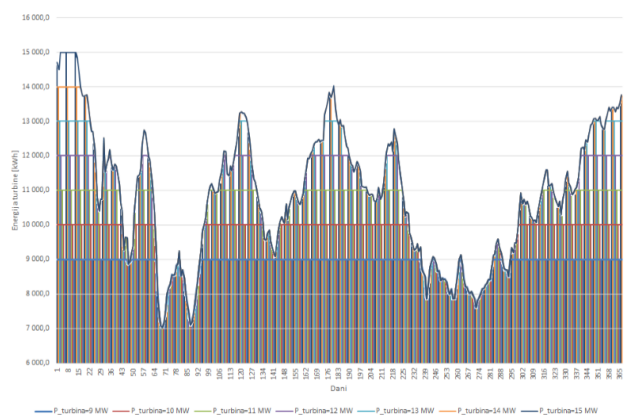
Koeficijent korisnog dejstva se izabere na osnovu tipa turbine. Osobine reke kod brane (nizak pad i velika količina vode) uzet je Kaplanov tip turbine sa

$\eta_{\text{turbina}}=0,85$. teoretska snaga turbine u zavisnosti od protoka po danu u godini prikazana na sledećoj slici (slika 3). Vidi se da ona varira u širokom opsegu, od 7 MW do 16 MW.

Međutim, proizvedena električna energija zavisi od veličine turbine, odnosno od nominalne snage odabranog generatora. Ako se izabere turbina-generator veće snage, tokom godine neće raditi punim kapacitetom. Ako se bira turbina-generator manje snage, dobar deo snage vode neće biti iskorišćen. Tada, se višak vode preko ventila mora odvesti u drugi kanal i tako poštediti turbinu od preopterećenja. U radu analizirana je turbina-generator sa snagom od 9 MW do 15 MW, u koracima od 1 MW. Na slici 4, predstavljeni su rezultati ove analize. Može se videti da je snaga turbine oko 12 MW najpovoljnija.



Slika 3. Grafički prikaz prosečne moguće izlazne snage na osovini turbine u periodu 2007-2017



Slika 4. Prikaz promene snage za optimalnu turbine

6. TEHNO-EKONOMSKA ANALIZA

Zavisno od uslova na mestu izgradnje, zakona u datoj državi itd. cena izgradnje male HE zavisi od:

- tipa male HE
- instalisane snage i broja generatora
- iskoristivog pada
- kapaciteta akumulacije
- lokacije

Nabavka i montaža generatora, automatike i ostali troškovi pribora koštaju najviše izavise od veličine generatora. Sva energija koja dolazi sa turbine ne može direktno da se pretvori u električnu energiju. Postoje mehanizmi između dva sistema. U radu turbine i generatora povezanog sa reduktorom, koji ima svoju

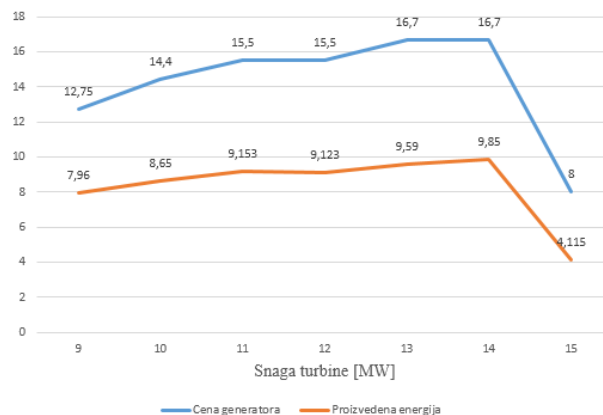
efikasnost ($\eta_{\text{reduktor}}=0,92$), koji se vidi u formuli za snagu generatora.

$$P_{\text{generator}} = \eta_{\text{reduktor}} \cdot \eta_{\text{generator}} \cdot P_{\text{turbina}} \quad (3)$$

$$P_{\text{generator}} = 0,92 \cdot 0,98 \cdot P_{\text{turbina}} \quad (4)$$

$$P_{\text{generator}} = 0,9016 \cdot P_{\text{turbina}} \quad (5)$$

U radu su analizirane sve snage generatora, sa kombinacijom snaga turbine, koje imaju vrednosti koje su ranije napomenute. Analiza isplativosti tokom izbora generatora i turbine fokusirana je na cenu generatora i proizvedene energije u godini. Na slici 5 prikazana je cena generatora (u milionima) i proizvedena energija (u MW) u zavisnosti od snage turbine.



Slika 5. Grafički prikaz proizvedene energije i cene generatora u zavisnosti snage turbine

Iz ovog aspekta, rezultati prikazuju da se najbolje i najbrže ostvaruje finansijsko vraćanje može očekivati kada se radi sa turbinom 12 MW. Kod ovakve koncepcije, s obzirom na gubitke na vratilu i prenosu energije do spojnice generatora, može proračunati da je potrebno izabrati nominalnu snagu generatora od 9 MW (preciznije 9.123 kW). U tom slučaju, povraćaj investicije je u roku od 3 godine.

Povećava se cena turbine, ako se radi sa pomerajućim lopaticama (*variable blades*). Sa fiksiranim lopaticama (*constant blades*) ne može da se upravlja količinom vode koja prolazi. Sa korišćenjem pomerajućih lopatica postiže se bolja iskoristivost vode koja prolazi kroz turbinu.

7. SOFTVERSKI ALATI ZA PROJEKTOVANJE MALIH HE

7.1. Softver RETScreen

RETScreen je programski paket koji je razvila Vlada Kanade i prva verzija je dostupna javnosti od 1998.godine. Trenutna verzija dostupna je od 2016. godine, kada je prezentovana u San Francisku na ministarskoj konferenciji o čistoj energiji. Softver omogućava sveobuhvatnu identifikaciju, procenu i optimizaciju tehničke i finansijske održivosti potencijalnih projekata iz oblasti obnovljive energije i energetske efikasnosti, kao i merenje i verifikaciju stvarnih performansi objekata i identifikaciju mogućnosti za uštedu/proizvodnju energije. Kada se podešava simulator za konkretni slučaj, namešta se da lokacija bude Novi Bečej, i izabere ponuđeni, odnosno postojeći

generator čija snaga iznosi 10 MW, dobija se da se isplaćuje u roku od 2,2 godine.

Financial viability			
Pre-tax IRR - equity	%	47,7%	
Pre-tax MIRR - equity	%	18,5%	
Pre-tax IRR - assets	%	15,4%	
Pre-tax MIRR - assets	%	11,6%	
Simple payback	yr	4,8	
Equity payback	yr	2,2	
Net Present Value (NPV)	€	20 675 491	
Annual life cycle savings	€/yr	2 264 927	
Benefit-Cost (B-C) ratio		5,3	
Debt service coverage		2,7	
GHG reduction cost	€/tCO ₂	-44	
Energy production cost	€/kWh	0,041	

Slika 6. Finansijska analiza sistema u RETScreen programu

7.2 Softver RS Minerve

Program koji se bavi hidrološko-hidrauličkim modelovanjem za simulaciju padavina i usmeravanje vodenog toka zove se *RS Minerve*. Prednosti u odnosu na ostale programe su da se gradi u različitim hijerarhijskim nivoima (tzv. pod-modeli). Korisnički prilagođeni interfejs je napravljen za lako razumevanje programa, ne mora da se kreiraju meteorološke serije za svaki element modela, već samo baza podataka sa posmatranim meteorološkim podacima itd.

7.3 Softver HOMER

HOMER (Hybrid Optimization of Multiple Electric Renewables) računarski program, tokom vremena se razvio do jednog od glavnih programa za optimizaciju dizajna mikro mreža. *HOMER* pokušava da simulira održivi sistem za sve moguće kombinacije opreme koju želite da razmotrite. U zavisnosti od toga kako je model postavljen, može simulirati stotine ili čak hiljade sistema. Sistem koji se simulira može da se sortira prema različitim kriterijumima koje definišete, tako da možete videti najbolja moguća rešenja. Program služi za analizu obnovljivih izvora. Pored toga može da prikazuje sačuvana goriva. Analiza osetljivosti je opcioni korak koji omogućava modelovanje uticaja promenljivih veličina koje su izvan vaše kontrole, kao što su brzina vetra i troškovi goriva, i vidi se kako se optimalni sistem menja s ovim varijacijama.

7.4. Primena softvera RETScreen za ME na brani

Pregledom mogućnosti tri programa, za modelovanje malih HE najpogodniji je programski paket *RETScreen*. Prvo se unesu premenljive programa, kao što su lokacija, objekat, itd. Pomoću ovih podataka, softver može da prikaže klimatske i vremenske uslove u podešenom mestu. *RETScreen Climate Database* sadrži meteorološke podatke koji su važni za model. Moguća snaga male hidroelektrane koja može da se dobije na osnovu date vrednosti protoka Q određuje se prema sledećoj jednačini, u kojoj su uvaženi hidraulični gubici i uticaj odvođenja vode:

$$P = \rho \cdot g \cdot Q \cdot [H_g - (h_{hydr} + h_{tail}) \cdot \eta_t \cdot \eta_g \cdot (1 - l_{trans}) \cdot (1 - l_{para})] \quad (6)$$

U simulatoru dobijeno je da generator od 10 MW sa unetom cenom električne energije, pokazuje da hidroenergetski sistem pokriva troškove izgradnje elektrane za 2,2 godine.

8. ZAKLJUČAK

MH imaju veliki potencijal razvoja u oblasti obnovljivih izvora energije, a njihov rad pomaže stabilizaciji i ekonomskom razvoju i ima minimalni doprinos emisiji CO₂.

U radu su opisani parametri hidroelektrane na brani kod Novog Bečaja. Pri izboru osobine turbine, potrebno je izabrati onu koja se najbolje uklapa u uslove na Tisi, radi visine pada vode i proticanjem vode koja se meri kod Novog Bečaja. Kroz rada može da se zaključi da brana ima mogućnost stavljanja Kaplanove turbine od 12 MW, i generator od 9.1 MW i to se najbrže isplati u roku od tri godine.

Za lakše projektovanje hidrosistema koriste se programski paketi, ali je neophodno koristiti različite programe, jer je svaki specijalizovan za određen aspekt sistema. Na temu hidroelektrana kod Novog Bečaja urađeno je više radova. Analiziran je vremenski period drugačije nego u ovom radu, pa stoga, naravno zaključci nisu isti, ali može da se kaže da su slični ovima u radu [9].

9. LITERATURA

- [1] Web stranica: Hydro Energy - Energija Vode, http://www.centrala.org.rs/knjiga%20liber%20perpetuum/04Energija_vode_Hydro_Energy.pdf
- [2] Web stranica: <http://www.vodevojvodine.com/pdf/Brochures/Knjiga%20blaga.pdf>
- [3] J. Savković "Mogućnosti primjene modela piko hidroelektrane", Master rad, FTN, Novi Sad, 2018.
- [4] Web stranica: <http://www.dlhe.rs/latinica/he-bajina-basta-1>
- [5] B. Lepotić Kovačević, B. Lazarević "Izgradnja postrojenja i proizvodnja električne energije u hidroelektranama do 30 MW u Republici Srbiji", Programa Ujedinjenih nacija za razvoj, Jun 2016.
- [6] Web stranica: <https://www.ems.rs/>
- [7] Web stranica: <http://www.vodevojvodine.com>
- [8] V. Mijailović "Distribuirani izvori energije- Principi rada i eksploatacioni aspekti", Akademski misao, Beograd, 2011.
- [9] D. Milićević, Lj. Anđelković, A. Živković "Mogućnosti iskorišćenja hidroenergetskog potencijala reke Tise na lokaciji brane kod Novog Bečaja", INFOTEH, Jahorina, Vol. 13, Mart 2014., pp.190-195, <https://infotech.rs.ba/zbornik/2014/radovi/ENS-2/ENS-2-12.pdf>

Kratka biografija:



Andor Rajšli rođen je u Senti 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine odbranio je 2019.god. kontakt: rajšli.andor@gmail.com



dr. Vladimi A. Katić, red. prof. rođen je 1954. godine u Novom Sadu. Doktorirao je na Univerzitetu u Beogradu 1991. godine. Oblasti interesovanja su mu energetska elektronika, obnovljivi izvori električne energije, kvalitet električne energije i električna vozila.

PRIMENA METODA JEDNOSTAVNOG EKSPONENCIJALNOG IZGLAĐIVANJA ZA PRORAČUN PROGNOZE PROTOKA PRIRODNOG GASA**IMPLEMENTATION OF SIMPLE EXPONENTIAL SMOOTHING METHOD FOR NATURAL GAS CONSUMPTION FORECASTING**

Luka Ivezic, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu se razmatra problem prognoze protoka prirodnog gasa, koji se zbog minimalnog zagađenja životne sredine danas sve više koristi za zagrevanje objekata, naročito stambenih. Prvo je prikazan teorijski uvod i data postavka problema, a zatim je formiran matematički model za prognozu zasnovan na metodu jednostavnog eksponencijalnog izgladivanja. Numerička verifikacija matematičkog modela izvršena je na jednostavnoj test mreži. U tu svrhu, prethodno formirani model za prognozu razvijen je u programskom jeziku C#. Dobijeni rezultati predstavljeni su kroz dijagrame na osnovu kojih je izvršena analiza.

Ključne reči: Prognoza protoka prirodnog gasa, Metod jednostavnog eksponencijalnog izgladivanja

Abstract – In this paper main focus is on forecast of natural gas consumption, which is nowadays increasingly used for heating objects, especially residential, because of its minimal contamination of natural environment. First, it is shown the theoretical introduction and given setting of the problem, then mathematical model for forecast based on simple exponential smoothing method was formed. Numerical verification of mathematical model is performed on simple test gas network. In that purpose, the model for forecast was developed in the C# programming language. The obtained results are presented through diagrams on the basis of which the analysis is performed.

Keywords: Forecast of the natural gas consumption, Simple exponential smoothing method

1 UVOD

U današnje vreme moderne evropske zemlje se uglavnom oslanjaju na prirodni gas kao primarni energent. Prirodni gas je izuzetno važan izvor energije zbog smanjenja zagađenja i održavanja čiste i zdrave životne sredine. Iako je obično grupisan sa drugim fosilnim gorivima i izvorima energije, postoje mnoge karakteristike koje ga čine jedinstvenim. To je jedan od najčistijih, najsigurnijih i najkorisnijih izvora energije. Pored toga, očekuje se da će svetske cene nafte ostati visoke, te će se prirodni gas koristiti gde god je to moguće [1]. Kompanije koje se

bave distribucijom gasa, nalaze se pred konstantnim izazovom da svojim korisnicima obezbede neprekidan i kvalitetan dotok prirodnog gasa. Za ostvarivanje ovog cilja neophodna je dobra gasna prenosna mreža. Ovakav kompleksan sistem se sastoji od [2]: gasnih stanica, kompresora, cevi, gasnih skladišta i naravno potrošača, povezanih na ovakav sistem. Da bi se obezbedio efikasan, ekonomičan i bezbedan dotok gasa, unapred se, sa visokom preciznošću moraju znati dnevne potrebe za gasom svih potrošača u mreži. Prognoza protoka gasa vrši se na bazi od jednog sata za naredna dva gasna dana [2]. Pod gasnim danom podrazumeva se period od 24 uzastopna časa, počevši od nekog predefinisano momenta u toku dana. Potrošači gas koriste najviše za zagrevanje vazduha – grejno opterećenje. Gas se koristi i u druge svrhe, kao što su zagrevanje vode, kuvanje i ostale procese – bazno opterećenje. Gas se može koristiti i kao energent iz kojeg se dobija električna energija [2].

U drugom poglavlju postavljen je problem koji će biti rešavan u radu – problem prognoze protoka prirodnog gasa za naredna dva gasna dana.

U trećem poglavlju formiran je matematički model, odnosno algoritam zasnovan na metodu eksponencijalnih pokretnih proseka [3,4]. Pokazano je kako se računa osnovna prognoza i njena greška [5]. Na kraju ovog dela je opisano dobijanje konačne, korigovane prognoze protoka prirodnog gasa.

U četvrtom poglavlju prikazani su rezultati proračuna dobijeni na primeru jednostavne test mreže [6]. Tačnost primenjenog algoritma određena je pomoću srednje apsolutne procentualne greške, tzv. *MAPE* (*Mean Absolute Percentage Error*) [7,8]. Na osnovu dobijenih rezultata vrši se analiza.

U petom poglavlju dat je zaključak o tome zašto je prognoza protoka prirodnog gasa važna, moguća unapređenja algoritma kao i pravci daljeg istraživanja u ovoj oblasti. Nakon zaključka referentno je navedena literatura korišćena za realizaciju ovog rada.

2 POSTAVKA PROBLEMA

Osnovni problem koji se u radu razmatra je problem prognoze potrošnje prirodnog gasa za narednih 48 sati, na bazi od jednog sata. Prognoza se obično vrši od trenutnog vremena, ali opciono se može vršiti od bilo kog trenutka za koji su dostupni istorijski podaci. Za svaki sat u posmatranom intervalu postoji po jedan model za prognozu. Svaki model zahteva, kao ulaz, vrednosti protoka i efektivne temperature za trenutni sat za sve

NAPOMENA:

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

gasne stanice u mreži. Ove vrednosti su potrebne kako bi se izvršilo prilagođavanje modela. Satna prognoza temperature na gasnoj stanici tokom perioda predikcije potrebna je da bi se proizvela prognoza potražnje gasa. U tu svrhu svaka gasna distributivna stanica ima pridruženu meteorološku stanicu. Vremenska prognoza koja se koristi u prognoziranju protoka prirodnog gasa može biti od trenutnog vremena, ili možemo koristiti arhivirane podatke u slučaju istorijskog predviđanja.

2.1 Adaptacija

Proces adaptacije se izvršava na kraju svakog intervala, kako za meteorološke stanice, tako i za gasne stanice.

2.1.1 Meteorološke stanice

Informacije o vremenu koje prima svaka meteorološka stanica koriste se za izračunavanje efektivne temperature na kraju intervala adaptacije.

2.1.2 Gasne stanice

Za svaku gasnu stanicu odvijaju se sledeći koraci pri izvođenju adaptacije ako su dostupni satni protok i efektivna temperatura:

- određuje se redukovana temperatura,
- izračunavaju se osnovna prognoza i njena greška,
- vrši se detekcija tačaka koje odstupaju od preostalih tačaka posmatranog skupa (eng. *outlier* – izuzetak, otpadak),
- ako su zadovoljeni odgovarajući kriterijumi adaptacije:
 - radni dani: prilagođavanje nagiba i pivotnih proseka,
 - dani vikenda: prilagođavanje pivotnih proseka i faktora vikenda,
 - kvazi-praznici: prilagođavanje pivotnih proseka,
 - praznici: nema adaptacije.
- ažuriranje vremena adaptacije, poslednje izračunate greške i korelacijskih proseka.

Konačna, korigovana prognoza protoka prirodnog gasa dobija se dodavanjem greške izračunate u istorijskoj adaptaciji na osnovnu prognozu protoka gasa.

3 FORMIRANJE MATEMATIČKOG MODELA

Matematički model za proračun prognoze protoka prirodnog gasa zasnovan je na metodu eksponencijalnih pokretnih proseka, odnosno metodu jednostavnog eksponencijalnog izgladivanja [3,4]. Jednačina kojom se opisuje ovaj metod ima oblik [3]:

$$F_{t+1} = \alpha \cdot y_t + (1 - \alpha) \cdot F_t, \quad (3.1)$$

gde je:

- y_t – stvarna, poznata serijska vrednost u trenutku t ,
- F_t – prognozirana vrednost varijable y u trenutku t ,
- F_{t+1} – prognozirana vrednost u vremenskom trenutku $t+1$,
- α – konstanta (faktor) izgladivanja.

Konstanta izgladivanja α je odabrani broj između nula i jedan, $0 < \alpha < 1$ [4]. Preuređivanjem formule (3.1) uočava se da je:

$$F_{t+1} - F_t = \alpha \cdot (y_t - F_t), \quad (3.2)$$

Dakle, promena u prognoziranoj vrednosti je proporcionalna greški prognoze. Izraz (3.2) se zapisuje i kao:

$$F_{t+1} = F_t + \alpha \cdot e_t, \quad (3.3)$$

gde je rezidual:

$$e_t = y_t - F_t, \quad (3.4)$$

greška predviđanja u trenutku t .

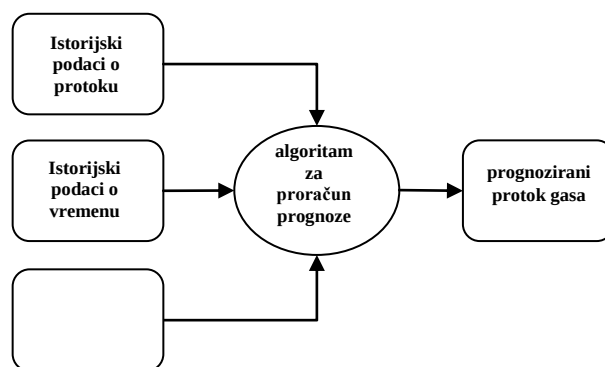
Korišćenjem jednačine (3.1) vrši se proračun pokretnih proseka vremenskih faktora – temperature, brzine vetra i stepena oblačnosti, potrebnih za dobijanje redukovane temperature, složene funkcije vremena. Pokretni proseci ove veličine i protoka gasa su u linearnoj korelaciji i određuju osnovnu prognozu protoka gasa. Parametri koji se uzimaju kao ulazni za algoritam zasnovan na metodu eksponencijalnih pokretnih proseka su:

- istorijski podaci o protoku gasa,
- istorijski podaci o vremenu za temperaturu, brzinu vetra i stepen oblačnosti,
- tipovi dana za istorijske datume i za dva dana unapred – razlikuju se radni dani, dani vikenda i praznični dani (pravi praznici i kvazi-praznici),
- vremenska prognoza za sledeća dva gasna dana za temperaturu, brzinu vetra i stepen oblačnosti,
- istorijski podaci o danima obeleženim kao neželjeni.

Pod neželjenim danima se podrazumevaju dani koji se žele isključiti iz skupa za trening. To su istorijski datumi čiji podaci nisu validni iz nekog od sledećih razloga [5]:

- neispravnost mernih instrumenata,
- ljudski faktor, greške prilikom očitavanja mernih instrumenata,
- podaci ili komponente agregiranih podataka nedostaju,
- veliki broj tačaka koje odstupaju od preostalih tačaka iz skupa podataka – otpatci, izuzeci.

Na slici ispod su prikazani ulazi i izlazi razmatranog algoritma:



Slika 1 – Algoritam za proračun prognoze [1]

4 PRIMER PRORAČUNA

Verifikacija algoritma za proračun prognoze protoka prirodnog gasa vrši se na jednostavnoj test mreži, koja je prikazana u nastavku. Originalni algoritam se modifikuje u cilju dobijanja tačnije prognoze. Razlikuju se pre svega sledeće varijante, algoritam kao ulaz koristi:

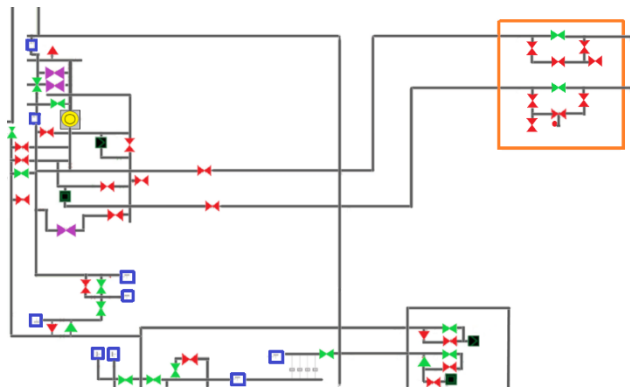
- podatke o protoku i vremenu iz skorije istorije, te se vrši predviđanje istorijske greške u cilju dobijanja konačne prognoze – originalni algoritam,

- podatke o protoku i vremenu sličnog dana iz istorije, te se vrši predviđanje istorijske greške u cilju dobijanja konačne prognoze – prvi predlog za unapređenje,
- podatke o protoku i vremenu iz skorije istorije, dok se na osnovnu prognozu dodaje apsolutna vrednost istorijske greške u cilju dobijanja konačne prognoze – drugi predlog za unapređenje,
- podatke o protoku i vremenu sličnog dana iz istorije, a na osnovnu prognozu se dodaje apsolutna vrednost istorijske greške u cilju dobijanja konačne prognoze – treći predlog za unapređenje.

Rezultati su dobijeni na osnovu matematičkog modela razvijenog u C# programskom jeziku.

4.1 Jednostavna test mreža

Posmatrana mreža se sastoji od velikog broja čvorova, svaki od čvorova predstavlja jednu distributivnu stanicu koja napaja određen broj potrošača. Rezultati proračuna će biti sakupljeni, prikazani i poređeni samo za jednu gasnu distributivnu stanicu. Zbog velike test šeme, na slici 2 prikazan je samo deo test mreže, gde je narandžastom bojom uokvirena distributivna gasna stanica koja se posmatra, a plavom bojom skupovi potrošača koji se napajaju sa obeležene stanice. Svetlo zeleni simboli predstavljaju otvorene ventile, crveni zatvorene (i jedni i drugi su mehanički), dok su ljubičasti ventili daljinski upravljivi (telemetrisani). Tamno zeleni kvadrati predstavljaju stanice za snižavanje pritiska. Žuti simbol označava kompresorsku stanicu za povećanje pritiska u mreži.



Slika 2 – Test mreža [6]

Zbog jednostavnijeg prikaza grafici će biti predstavljeni samo za jedan od datuma prikazanih u tabeli 1, i to za 14.12.2015 – ponedeljak, radni dan u zimskoj sezoni. Tačnost algoritma za prognozu protoka prirodnog gasa kvantifikovana je na osnovu srednje apsolutne procentualne greške (MAPE) [7,8].

Ova greška se računa na sledeći način:

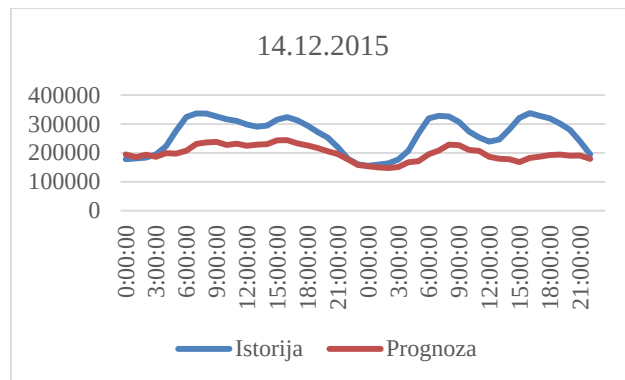
$$MAPE = \frac{100}{n} \cdot \sum_{t=1}^n \left| \frac{A_t - F_t}{A_t} \right|,$$

gde je:

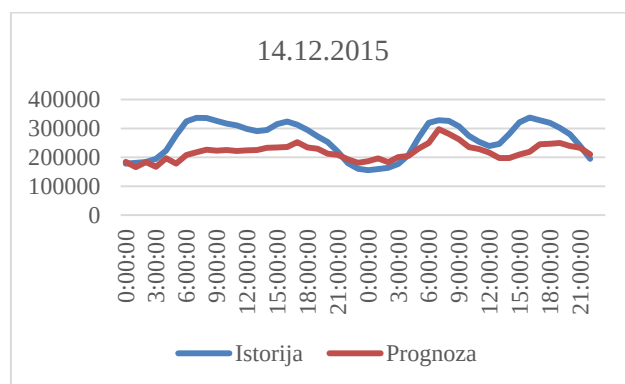
- n – broj sati za koje se MAPE računa,
- A_t – stvarne vrednosti protoka (istorijski podaci),
- F_t – izračunate vrednosti (primenom algoritma).

Prema vrednosti MAPE, razlikuju se četiri nivoa tačnosti prognoze. Da bi se prognoza smatrala odlično tačnom,

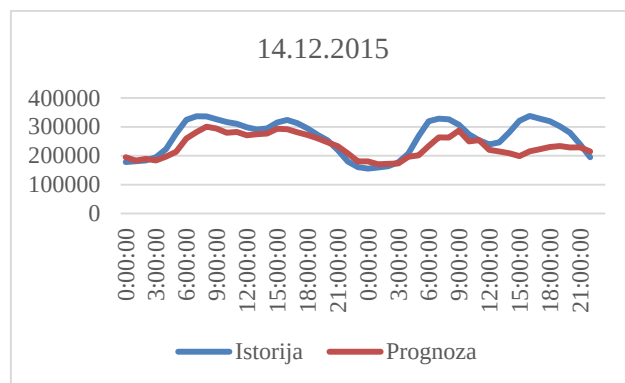
mora da bude zadovoljen kriterijum da je izračunata vrednost MAPE manja od 10%, između 10÷20% prognoza je dobra, između 20÷50% prihvatljiva i preko 50% MAPE znači da je prognoza netačna [3].



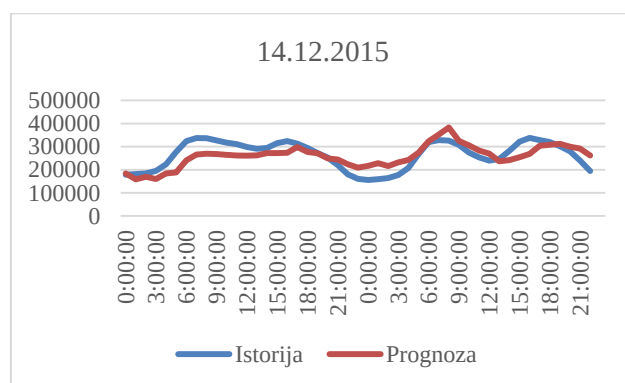
Slika 3 – Rezultati proračuna – originalni algoritam



Slika 4 – Rezultati proračuna – prvi predlog



Slika 5 – Rezultati proračuna – drugi predlog



Slika 6 – Rezultati proračuna – treći predlog

U nastavku su u tabeli 1 prikazane srednje apsolutne procentualne greške po intervalima, za sve datume od interesa. Za svaki algoritam su izračunate srednja i maksimalna greška u svakom od intervala, radi lakšeg poređenja rezultata.

Tabela 1 – MAPE po intervalima, za originalni algoritam, prvi, drugi i treći predlog za njegovo unapređenje

	MAPE ₆ [%]				MAPE ₁₂ [%]			
	o.a. ¹	p.p. ²	d.p. ³	t.p. ⁴	o.a.	p.p.	d.p.	t.p.
04.05.'15	11.2	26.2	10.1	14.4	9.4	31.6	13.1	9.2
16.09.'15	20.7	14.5	27.3	13.7	32.9	8.9	33.2	8.1
14.12.'15	10.1	12.2	8.9	15.2	29.6	32.2	12.9	19.5
19.03.'16	16.6	7.2	16.8	8.0	28.1	31.9	15.8	17.6
11.06.'16	21.5	9.7	20.4	9.9	33.5	20.3	24.7	13.4
Total	16.1	13.9	16.7	12.3	26.7	25.0	19.9	13.5
Max	21.5	26.2	27.3	15.2	33.5	32.2	33.2	19.5
	MAPE ₂₄ [%]				MAPE ₄₈ [%]			
	o.a.	p.p.	d.p.	t.p.	o.a.	p.p.	d.p.	t.p.
04.05.'15	9.6	22.6	39.9	37.8	23.9	28.1	66.7	16.9
16.09.'15	15.6	13.4	14.2	11.3	9.4	15.4	7.6	21.6
14.12.'15	18.0	18.1	8.0	11.4	26.8	16.8	16.4	16.0
19.03.'16	31.4	42.9	17.3	14.6	24.8	28.5	23.1	25.0
11.06.'16	31.0	20.9	13.7	11.4	30.6	14.6	11.1	10.8
Total	21.1	23.6	18.6	17.3	23.1	20.7	25.0	18.1
Max	31.4	42.9	39.9	37.8	30.6	28.5	66.7	25.0

Detaljnou analizom i podelom datuma po sezoni, a zatim i po tipu dana, zapaža se sledeće:

- u sezoni leto, najbolje rezultate daje treći predlog za unapređenje originalnog algoritma; dosta dobre rezultate daje prvi predlog;
- u sezoni zima, najbolji je drugi predlog za unapređenje; u stopu ga prati treći predlog čija je greška prognoze u toj sezoni malo veća;
- za kvazi-praznik, najbolju prognozu daje originalni algoritam; prihvatljive rezultate daje i treći predlog;
- najbolju prognozu protoka gasa za radne dane daju algoritmi koji predstavljaju drugi i treći predlog za unapređenje;
- za vikende najbolje rezultate daje treći predlog za unapređenje.

Na osnovu date analize, za sva četiri vremenska intervala, uočava se da je najbolji poslednji, treći predlog za unapređenje algoritma.

5 ZAKLJUČAK

Izazov koji se postavlja pred distributere, da na vreme dostave dovoljne količine kvalitetnog prirodnog gasa bez prekida, svakim danom je sve veći. Da bi se izbegli viši

operativni troškovi i plaćanje novčanih naknada, važno je da prognoza potražnje prirodnog gasa bude što tačnija. Originalni algoritam za proračun prognoze zasnovan na metodu jednostavnog eksponencijalnog izgladivanja dosta greši, te su data tri predloga za njegovo unapređenje. Od pomenutih predloga najbolje rezultate daje treći predlog za unapređenje. Međutim, bez obzira o kojoj verziji algoritma je reč, srednja apsolutna procentualna greška daleko je veća od praktično prihvatljivih 5÷10%. Takođe, maksimalna greška u pojedinim intervalima premašuje 40% ili čak 50%, što dovodi do zaključka da metod eksponencijalnih pokretnih proseka (jednostavnog eksponencijalnog izgladivanja) u oblasti prognoze praktično nije upotrebljiv.

6 LITERATURA

1. N.Aras: Forecasting Residential Consumption of Natural Gas Using Genetic Algorithms; *Energy Exploration & Exploitation*, 2008, Vol. 26, No. 4, pp. 241-266
2. S.R.Vitullo, R.H.Brown, G.F.Corliss, B.M.Marx: Mathematical Models for Natural Gas Forecasting; *Canadian Applied Mathematical Quarterly*, University of Alberta, Alberta, 2009, Vol. 17, No. 4, pp. 807-827
3. E.Ostertagova, O.Ostertag: The Simple Exponential Smoothing Model, *The 4th International Conference on Modelling of Mechanical and Mechatronic Systems*, Košice, September 2011, pp. 380-384
4. E.Ostertagova, O.Ostertag: Forecasting using simple exponential smoothing method; *Acta Electrotechnica et Informatica*, 2012, Vol. 12, No. 3, pp. 62-66
5. H.N.Akouemo, R.J.Povinelli: Probabilistic anomaly detection in natural gas time series data; *International Journal of Forecasting*, July-September 2016, Vol. 32, No. 3, pp. 948-956.
6. W.A.W.Abu Bakar, R.Ali: *Natural Gas*, Department Of Chemistry, University Teknologi Malaysia, Skudai, Johor, Malaysia, August 2010, Page 6.
7. S.Kim, H.Kim: A new metric of absolute percentage error in intermittent demand forecasts; *International Journal of Forecasting*, July-September 2016, Vol. 32, No. 3, pp. 669-679
8. L.Y.Ren: Revised Mean Absolute Percentage Errors (MAPE) on Errors from Simple Exponential Smoothing Methods for Independent Normal Time Series; *The Journal of American Academy of Business*, 2007, Vol. 10, No. 2, pp. 65-70

Kratka biografija:



Luka Ivezić rođen je u Nevesinju, BiH, 08.04.1994. godine. Osnovne akademske studije završio na Fakultetu tehničkih nauka – oblast Elektrotehnika i Računarstvo, smer Elektroenergetski sistemi 2017. godine i iste godine upisao master studije na istom fakultetu.

¹ originalni algoritam

² prvi predlog

³ drugi predlog

⁴ treći predlog

USREDNJNI MODEL FN INVERTORA SA NAPREDNIM UPRAVLJANJEM AVERAGED MODEL OF PV INVERTER WITH ADVANCED CONTROL

Jovana Marković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu je prikazan usrednjeni model FN invertora sa naprednim upravljanjem. Prikazani su načini modelovanja kao i rezultati simulacije modela u realnom vremenu koristeći Typhoon HIL uređaj.

Ključne reči: Usrednjeni model, FN inverter, Simulacija sistema, Modelovanje u realnom vremenu

Abstract – In this paper averaged model of a PV inverter with advanced control is explained. Methods of modelling, as well as simulation results in real-time using the Typhoon HIL simulator are given.

Keywords: Power Electronics, Averaged model, PV inverter, Real time simulation.

1. UVOD

Prilikom modelovanja i kontrole nekog ispravljača obično je praksa da se krene od usrednjenog modela i da se u tom modelu izvrši strujna i naponska regulacija, da se odrede parametri regulacije, kao i da se postave odgovarajući filteri, da bismo te parametre kasnije mogli primeniti na prekidački model. Ono što je prednost usrednjenog modela u odnosu na prekidački jeste:

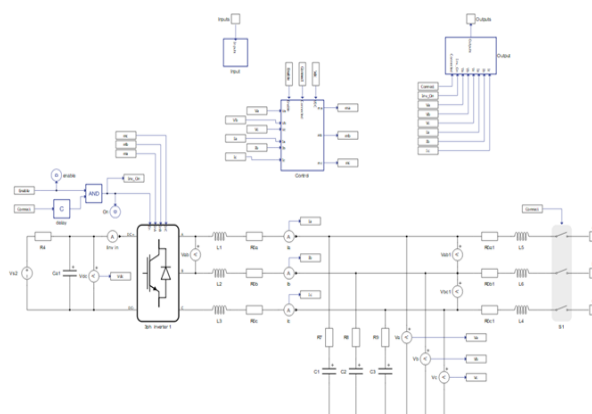
- Dosta brža simulacija sa stanovišta kompjutera,
- Jeftinija u odnosu na prekidački model (prekidačke komponente su jako skupe),
- Talasni oblici struje su čistiji u odnosu na prekidački model
- Ne gubi se na kvalitetu rezultata simulacije

U ovom radu je predstavljen metod usrednjavanja, kao i verifikacija rezultata usrednjenog u odnosu na prekidački model.

2. USREDNJNI MODEL FN INVERTORA

U današnjoj distributivnoj mreži i mikromreži povezani su sve više i više pametnih pretvarača - konvertora. Razvijanje i testiranje kontrole za ovaj veliki i složeni sistem zahteva nove simulacijske alate. Detaljna simulacija pretvarača je skupa, dok se sa drugim, usrednjenim, načinom modelovanja mogu postići dovoljno dobri rezultati. Cilj ovog rada je istražiti mogućnost zamene prekidačkog modela pretvarača sa naponom iza reaktanse i/ili kontrolisanim izvorom napajanja. Prilikom ovog postupka kontrola treba da bude

sa minimalnim brojem izmena. Sastavni deo ovakvih kontrolera je prekidač, koji je prilično skup, cilj ovog rada je bio da se ovaj prekidač zameni sa kontrolisanim naponskim izvorima pri čemu bi kontrola trebalo da bude nepromenjena.



Slika 1. Izgled modela sa prekidačkim upravljanjem [1].

Model se sastoji od DC dela sa krajnje leve strane, zatim invertora, koji je ovde najbitnija komponenta, LCL filtra i na kraju imamo mrežu. Jednosmerni deo čine kondenzator i naponski izvor, kao i merač napona i struje. Zadatak je bio da se zameni ovaj inverter sa kontrolisanim naponskim signalima i u zavisnosti od modela da izmodeluje DC deo prilagodjen tom primeru, a da se pri tome kontrola ne menja. Modelovanje ovog pretvarača sastoji se iz četiri koraka, koja su navedena u nastavku:

- Ispitati kako kontrolom međufaznih napona V_{ab} , V_{bc} , V_{ca} može da se utiče na mrežne struje u stacionarnom stanju. U ovom delu su ispisane jednačine Kirhofovih zakona. Za rešavanje ovih, kao i svih ostalih jednačina, kao pomoć korišćen je Matlab.
- Modelovanje LCL filtra
- Treći korak je dinamika sistema
- Modelovanje DC linka

2.1. Karakteristike FN invertora

Invertori su sistemi energetske elektronike, koji jednosmerni napon ili struju pretvaraju u naizmenični napon ili struju. Prema prirodi ulazne promenljive mogu biti naponski (VSI - voltage source inverters) ili strujni (CSI - current source inverters) invertori. Prema broju faznih priključaka na izlazu, invertori su najčešće monofazni ili trofazni, ali za pogon motora postoje

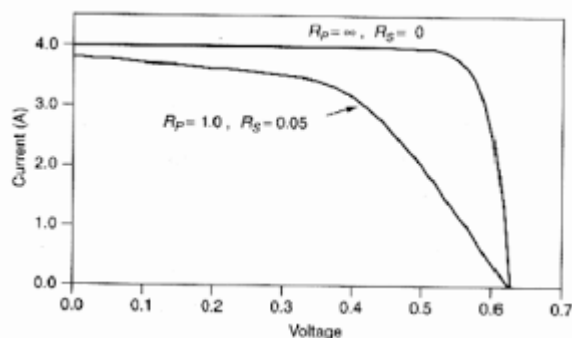
NAPOMENA:

Ovaj rad proistekao je iz master rada čiji je mentor bio dr Vladimir Katić, red.prof.

invertori i sa drugačijim brojem faza. Od invertora se zahteva da konverziju energije ostvare sa visokim koeficijentom korisnog dejstva, pa su na raspolaganju za sintezu invertora prekidači i reaktivni elementi. Izlazna veličina često treba da bude sinusoidalnog talasnog oblika, ali ponekad to nije slučaj. Sinusoidalni talasni oblik nije moguće bez disipacije stvoriti od napona ili struja konstantnih u vremenu, pa se zahtev za sinusoidalnim oblikom svodi na sinusoidalni oblik srednje vrednosti izlazne veličine tokom periode prekidanja. Stoga se usrednjavanje često koristiti prilikom analize invertora, što na nivou periode prekidanja, što na nivou periode modulišućeg signala, koja je obično znatno veća od periode prekidanja, pa se ovo usrednjavanje svodi na određivanje jednosmerne komponente DC napona [1]. Solarni invertori pretvaraju jednosmerni napon solarnih modula u naizmenični napon odgovarajuće faze i frekvencije sinhronizovan sa naponom mreže. Invertori se sa stanovišta regulacije dele na:

- Naponsko kontrolisani
- Strujno kontrolisani

Kada je reč o FN invertorima uglavnom se predstavljaju UI krivom iz koje se mogu pročitati podaci za modelovanje ovakve vrste invertora.



Slika 2. UI karakteristika FN panela [2]

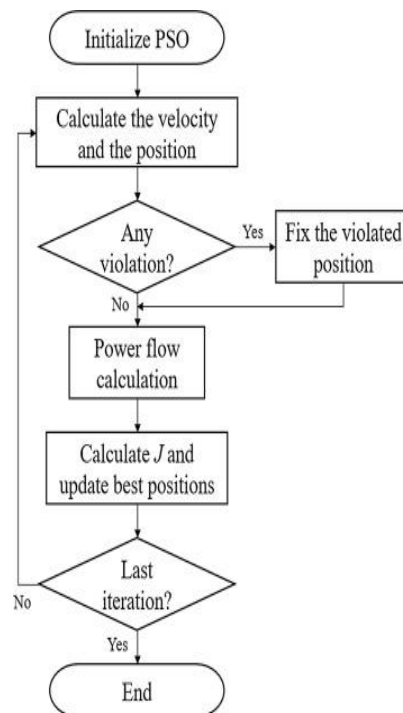
2.2. Uslovi priključenja FN invertora na mrežu

Grid kod podrazumeva uslove priključenja FN invertora na mrežu. U Nemačkoj je poznat BDEW standard, dok u Americi UL1741. Što se tiče BDEW kod njih je kriterijum sledeći: Ako je deo relevantne srednje naponske ili prenosne mreže višeg nivoa privremeno preopterećen, operator distribucijske mreže bi trebao biti sposoban da ograniči moć decentralizovanih postrojenja za proizvodnju električne energije u većem stepenu od 10% P_{nom} . Da bi to uradio, operator obično šalje kontrolni signal koji se obrađuje shodno tome i mora se implementirati kao ograničenje aktivne snage (tipično 60, 30 ili nulti procenat nominalne snage). Odgovarajuće potrebno ograničenje mora izvršiti inverter u roku od 60 sekundi. Frekvencija mreže naizmenične struje je konstantna u strogim granicama, u Evropi tačno 50 Hz. Napon se mora držati unutar određenih granica za sve nivoje mreže - posebno u distributivnoj mreži. Međutim, povećanje napona može se desiti tamo gde imamo povećanje aktivne snage na nisko naponskim i srednje naponskim nivoima, koji čine vezu dodatnih energetskih postrojenja više teško ili nemoguće. Pored toga, postojeća

fazna smena u procentima reaktivne snage u mreži smanjuje svoj prenosni kapaciteti povećava akumulisane gubitke prenosa. Tipični uzroci faznih smena su transformatori, veliki motori ili duže kablovske rute. Invertori sa reaktivnom snagom mogu pomoći nadoknađuje ravnotežu reaktivne snage u mreži ili da drže napon stabilan na mreži, da bi se obezbedio kvalitet napona u skladu sa standardom EN 50160 na taj način. Shodno tome, regulativa za srednji napon zahteva proizvodnju energije tako da se mogu snabdevati ili apsorbovati obe, i aktivna i reaktivna snaga. Operator mreže može zahtevati faktor snage $\cos(\varphi)$ između 0.95 i 1 [2].

3. LCL FILTER

Filter je neophodno staviti između VSI invertora i mreže, nametajući struju performanse za kontrolu povratne sprege i smanjenje harmonika izlazne struje. Može se koristiti jednostavan filter koji se sastoji od samo jednog kalema povezanog na red, ali harmonično slabljenje nije izraženo. Uglavnom se koristi LCL filter visokog reda konvencionalnog L-filtera za izravnavanje izlaznih struja iz VSI. LCL filter postiže veće slabljenje uz uštedu troškova, s obzirom na ukupnu težinu i smanjuje veličine komponenti. LCL filteri su korisni se u mrežnim priključcima i PWM ispravljačima, jer smanjuju količinu trenutne distorzija ubačene u komunalnu mrežu. Da bi se dizajnirao efikasan LCL filter, potrebno je imati odgovarajući matematički model filtera. Na slici 3. prikazan je algoritam filtera koji je korišćen u ovom radu. Prilikom izrade FN invertora sa naprednim upravljanjem za potrebe ovog rada, pasivno rešenje za prigušivanje je usvojeno, ali i druga rešenja se mogu aktivno primenjivati [3].



Slika 3. Algoritam LCL filtra [4]

4. MODELOVANJE DC LINKA

DC link je zapravo samo deo između ispravljača i invertora u opremi za konverziju energije u čvrstom stanju. Svrha DC konverzije je zapravo ta da nam je lakše upravljati DC talasima, koji su u suštini ravna linija nego talasima, koji su u obliku sinusoide. Neka definicija DC linka bila bi sledeća: DC link je metoda koja služi za povezivanje dva AC sistema koji imaju različite napone i frekvencije preko srednje DC faze. Nakon što se snaga konvertuje u DC, ona se čuva u kondenzatoru dok se konvertuje u AC [5]. Karakteristike kondenzatora DC linka su:

- visok napon (veći od 400 V DC),
- manji faktor disipacije,
- nisku induktivnost.
- formule za energiju.

DC link u ovom modelu je iz modelovan preko formule za energiju:

$$\frac{dW}{dt} = P_{in} - P_{out} \quad (1)$$

$$W = \frac{1}{2} C * V_{dc}^2 \quad (2)$$

$$i = C * \frac{dV}{dt} \quad (3)$$

Imamo dve snage, snagu sa FN panela i snagu mreže koje ulaze u sabirač, zatim imamo integrator koji je zamenio kondenzator i dalje preko formule za energiju smo došli do Vdc napona, koji dalje ulazi u kontrolu koja je ostala nepromenjena.

5. REZULTATI SIMULACIJE

U tabeli 1. prikazani su parametri koji su korišćeni prilikom modelovanja usrednjenog modela FN invertora sa unapredjenom kontrolom.

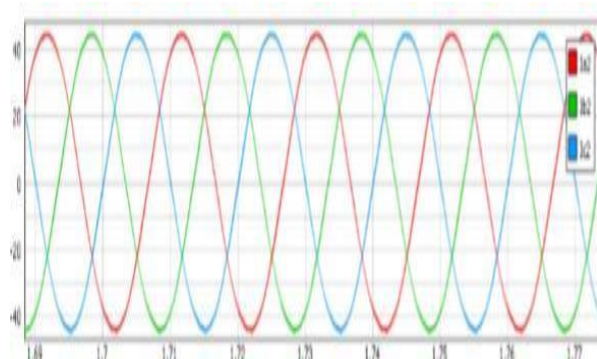
Tabela 1. Parametri simulacije

Nominalni napon	480 [V]
Nominalna snaga	5e6 [VA]
Frekvencija	60 [Hz]
Prekidačka frekvencija	10000 [Hz]
Napon DC linka	800 [V]
Površina panela	5556 [m ²]
Efikasnost panela	20 [%]
Korak simulacije	100e-6 [s]

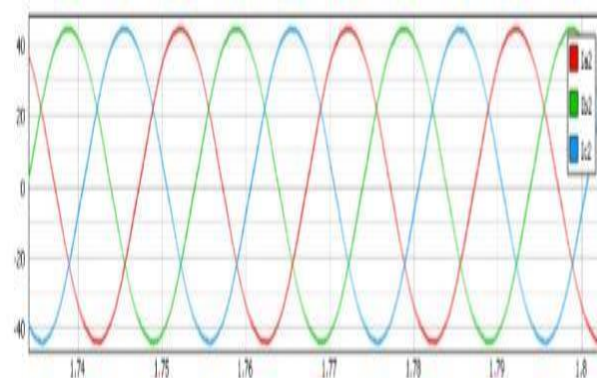
5.1. Poređenje rezultata simulacije u usrednjenom i u prekidačkom modelu

Kada uporedimo rezultate simulacije u usrednjenom modelu sa rezultatima iz prekidačkog modela vidimo da kada prekidače, koji su skupi i unose šumove u talasni oblik, zamenimo kontrolisanim naponskim izvorima, dobijamo potpunu istu funkcionalnost invertora. To vidimo iz cinjenice da smo pri istoj kontroli dobili potpuno iste amplitude u oba modela. Takođe vreme

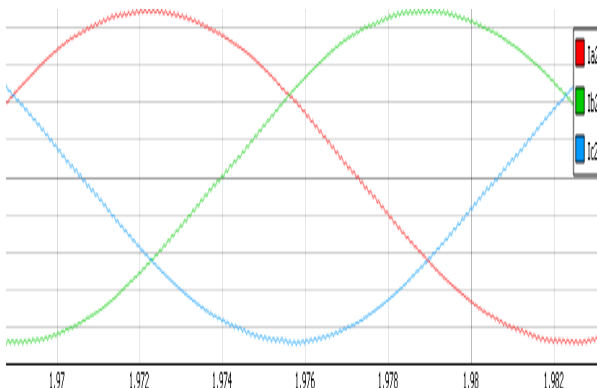
potrebno za izvršavanje simulacije znatno je redukovano u usrednjenom modelu u odnosu na prekidački model.



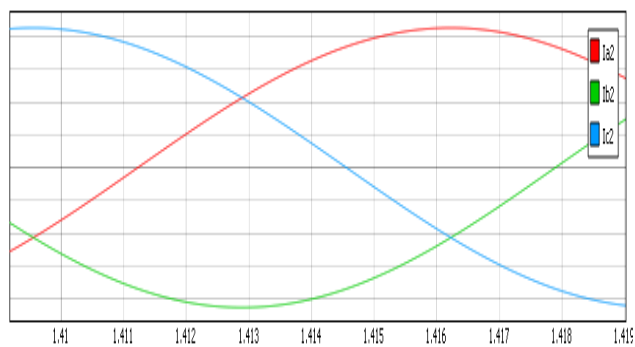
Slika 4. Izgled faznih struja u usrednjenom modelu



Slika 5. Izgled faznih struja u usrednjenom modelu



Slika 6. Uvećan prikaz talasnih oblika faznih struja u prekidačkom modelu



Slika 7. Uvećan prikaz talasnih oblika faznih struja u usrednjenom modelu

6. ZAKLJUČAK

Rad opisuje način modelovanja i simulacije u realnom vremenu usrednjenog FN invertora. U ovom radu su objašnjene osnove FN invertora, kao i simulacija u realnom vremenu. Opisane su ostale topologije i načini rada, kao i mane i prednosti svake. Rečeno je nešto više o samom Virtual HIL hardveru i sftoveru kao i njihovim mogućnostima. Testiranje i simulacija u realnom vremenu postaje sve bitnija u razvoju svakog projekta jer omogućuje bolje saznanje i ponašanje samog sistema i pre njegove i mplementacije u realnom sistemu. Ovaj rad pokazuje mogućnosti PV invertora kao i njeno priključenje na mrežu. Simulacija nam omogućava da sa velikom preciznosti pratimo vrednosti svih važnih signala.

7. LITERATURA

- [1] <http://www.cei.upm.es/master-on-industrial-electronics/master-theses/>
- [2] J. Kolar and T. Friedli, "The essence of three-phase pfc rectifier systems part i," Power Electronics, IEEE Transactions on, vol. 28, no. 1, pp. 176– 198, Jan2013
- [3] https://www.google.com/search?q=uv+kriva+pv+panela&rlz=1C1CHZL_enRS836RS836&oq=uv+kriva+p&aqs=chrome.1.69i57j69i59.6506j0j9&sourceid=chrome&ie=UTF-8
- [4] https://standardscatalog.ul.com/standards/en/standard_1741_2
- [5] https://www.google.com/search?q=algoritam+lcl+filtera&rlz=1C1CHBF_enRS839RS839&tbm=isch&source=iu&ictx=1&fir=QKxxKpCsM9OM%253A%252CHL7PVmCHI5BkM%252C_&vet=1&us

Kratka biografija



Jovana Marković rođena je u Loznici 1994. god. Gimnaziju je završila u Loznici, 2013 god. je upisala Fakultet tehničkih nauka, studijski program Energetika, elektronika i tele-komunikacije. Na studijama se opredelila za smer Elektroenergetika Elektroenergetski sistemi i diplomirala 17. 10. 2017. god. Na master studijama se opredelila za smer Distribuirani elektroenergetski resursi.

**PREGLED SOFTVERSKIH ALATA ZA SIMULACIJU I ODABIR PARAMETARA
TOPLOTNIH PUMPI****OVERVIEW OF SOFTWARE TOOLS FOR SIMULATION AND SELECTION OF HEAT
PUMP PARAMETERS**

Anja Subotin, Vladimir Katić, *Fakultet tehničkih nauka, Novi Sad*

Elektrotehnika i računarstvo – Energetska elektronika i električne mašine

Kratak sadržaj – U ovom radu opisano je nekoliko odabranih softverskih alata za simulaciju i odabir parametara toplotnih pumpi. Svaki od alata je ocenjen prema određenim kriterijumima, te je na kraju odabran softver koji je optimalan za simulaciju rada sistema. On je primenjen za simulaciju integracije toplotne pumpe u sisteme za skladištenje tople i hladne vode i za simulaciju različitih kombinacija performansi toplotne pumpe voda-voda.

Ključne reči: Obnovljivi izvori energije, geotermalna energija, toplotne pumpe, softverski alati za simulaciju i odabir parametara toplotnih pumpi.

Abstract –The paper describes several selected software tools for simulation and selection of heat pump parameters. 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 integration of the heat pump into a hot and cold water storage systems and to simulate various combinations of water-water heat pump performance.

Keywords: Renewable energy sources, geothermal energy, heat pump, software tools for simulation and selection of heat pump parameters.

1. UVOD

Poslednjih godina mnogo se govori o potrebi smanjenja potrošnje energije, odnosno povećanja energetske efikasnosti. Razlog tome je ubrzan rast tražnje vezan za globalni rast stanovništva i ubranu industrijalizaciju manje razvijenih zemalja, kao na primer Indije, Vijetnama, Nigerije, Angole i dr.

Rezerve fosilnih goriva su ograničene i troše se takvim tempom da se može očekivati njihovo iscrpljivanje već tokom ili najduže do kraja ovog veka. Istovremeno, njihovim korišćenjem se emituju razne čestice i gasovi staklene bašte, što ugrožava životnu sredinu i dovodi do pojave klimatskih promena. Iz tog razloga su u svetu pokrenut niz akcija za primenu tehnologija iskorišćavanja obnovljivih izvora energije, kao što su: energija biomase, solarna energija, energija vetra, vode i geotermalna energija.

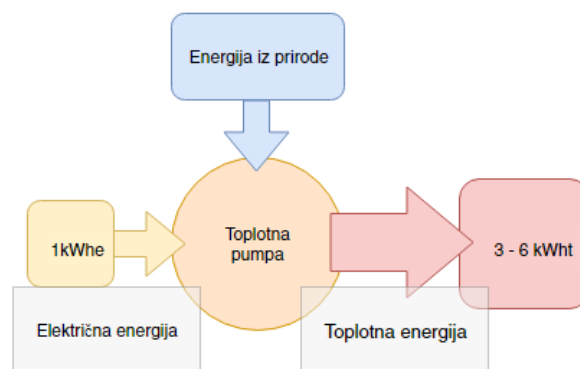
NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Katić, red.prof.

Jedan od načina za efikasnu upotrebu energije iz prirode, jeste primena geotermalne energije kroz tehnologiju toplotnih pumpi. Zahvaljujući najsavremenijim rešenjima, ovu energiju moguće je iskoristiti za pripremu vode ili vazduha za grejanje, a ujedno i za hlađenje. Efikasnost geotermalnog grejanja je znatno veća od standardnih oblika grejanja. Na primer, geotermalna energija je 48% efikasnija od kotlova na gas i više od 75% od peći na lož ulje. Njenim iskorišćavanjem ispušta se vrlo malo štetnih gasova u atmosferu, pa je ovaj vid grejanja/hlađenja ekološki vrlo prihvatljiv [1].

Princip rada toplotnih pumpi predstavljen je na slici 1. Putem izmenjivača toplote moguće je efikasno iskoristiti toplotnu energiju zemlje uz mali utrošak električne energije. Na taj način se može obezbediti oko 80% potreba objekta, dok se 20% mora utrošiti u vidu električne energije.

Pravilan odabir odgovarajućeg sistema i veličine toplotne pumpe veoma je važna faza u projektovanju ovih instalacija. On treba da ekonomično zagreva/hladi kuću tokom godine, ali i da se izbori sa najhladnijim i najtoplijim danima. Veća toplotna pumpa ima veće investicione troškove, ali i manje troškove rada, kao i obrnuto. Zato izbor veličine toplotne pumpe treba da da optimalni balans između investicionih i radnih troškova.



Slika 1. Princip rada toplotne pumpe.

Današnji softverski alati omogućavaju ovu optimizaciju i znatno olakšavaju i ubrzavaju proces odabira adekvatnog rešenja i projektovanje kompletnih sistema. Cilj ovog rada je da predstavi pregled svremenih softvera, koji su raspoloživi na tržištu. U obzir su uzeti sledeći softverski alati: GLD (Ground Loop Desing) softver, GeoT* SOL, Easy RSS, ETU softver, Solkane i TRNSYS. Takođe, cilj rada je i da preloži kriterijume za njihovo poređenje, te izvrši odabir najpovoljnijeg.

2. PREGLED SOFTVERSKIH ALATA

U ovom poglavlju će biti ukratko predstavljene osnovne karakteristike pomenutih softverskih alata za proračun i projektovanje sistema grejanja/hlađenja korišćenjem geotermalne energije na bazi tehnologije toplotnih pumpi.

2.1. GLD (Ground Loop Design) softver

Ground Loop Design (GLD) je vodeći softverski paket za projektovanje geotermalnih sistema toplotnih pumpi. Koristi se u više od 70 zemalja. Ovaj softver nudi set alata za brzo i automatsko dizajniranje cevnih sistema. Uključuje dva modula koji se odnose na dizajn cevovoda, a to su modul Piping Sistem Builder (PSB) i modul Computational Fluid Dynamics (CFD). Ova dva modula za projektovanje cevovoda bez problema se integrišu sa modulima dizajna izmenjivača toplote. Putem ove integracije, potrebno je samo nekoliko minuta da se dizajnira, optimizuje i odredi kompletan komercijalni cevovodni sistem [2].

Model toplotne pumpe koji se koristi u GLD programu reprodukuje kompletne operativne podatke bilo koje pojedinačne jedinice kada se isporučuje sa nekoliko reprezentativnih tačaka podataka izabranih iz celog interesnog područja. Podaci za svaku pumpu se mogu uneti u model i grupisati zajedno pod naslovom proizvođača i serije. Takođe, podaci se moraju uneti samo jednom, a zatim se mogu više puta koristiti za naredne sesije modelovanja. Mnoge popularne pumpe većih proizvođača su već uključene u program.

Neke od prednosti koje GLD softver pruža dizajnerima jesu: tačnost kalkulacije, pouzdan dizajn, lakoća korišćenja, brza iteracija i optimizacija, sveobuhvatna fleksibilnost i globalna podrška [2].

2.2. GeoT* SOL softver

GeoT* SOL je softver koji omogućava korisniku da bira između različitih izvora toplote i konfiguracija sistema. Ključna karakteristika je integracija solarnih termalnih sistema za toplu vodu za domaćinstvo ili grejanje prostora. Zahtevi, gubici i potrošnja se određuju kao rezultat dinamičke simulacije u minuti. Oni čine osnovu na kojoj se računaju potrošnja električne energije, godišnji koeficijent performansi i troškovi, uzimajući u obzir vanredne periode i tarife. Sa ovim parametrima GeoT* SOL softver procenjuje efikasnost sistema.

Uz pomoć ovog softvera, moguće je simulirati postojeće sisteme sa kotlom, gde je dodata toplotna pumpa. Nova postrojenja sa kotlom kao rezervom se uz pomoć GeoT* SOL softvera mogu prikazati i optimizovati. Način rada se može izabrati i parametrizovati za simulaciju iz velikog broja mogućih kombinacija monovalentnih, monoenergetskih i bivalentnih sistema sa paralelnim, delimično paralelnim ili alternativnim radom [3].

Sa GeoT * SOL softverom moguće je simulirati pet različitih tipova sistema toplotnih pumpi, od jednostavnih do vrlo složenih, svaki za konfiguraciju sa geotermalnim sondama za grejanje, vazduhom, geotermalnim kolektorima ili podzemnim vodama [3].

2.3. Easy RSS softver

Easy RSS je jedan od vodećih grupa u Velikoj Britaniji, koja nude softverska rešenja za obnovljive izvore

energije. Ovaj softver za toplotne pumpe uključuje: dimenzionisanje toplotne pumpe, odabir pomoćnog ulaza, ishod sertifikata o usaglašenosti, proračun buke, dimenzionisanje uzemljenja, izračunavanje tekućih troškova, proračun kompenzacija za korišćenje sistema grejanja na bazi obnovljivih izvora u sklopu Renewable Heat Incentive (RHI) u Velikoj Britaniji [4].

Softver za toplotne pumpe nudi mogućnost korisniku da odredi veličine cevi na zemlji, proračuna buku i izračuna troškove. Pored područja za dizajniranje toplotnih pumpi, Easy RSS softver omogućava i proračun gubitaka toplote, emitera i kalkulaciju tople vode [4].

2.4. ETU softver

ETU Softver razvija, prodaje i podržava praktične softverske alate u oblasti inženjeringa, simulacije, verifikacije usklađenosti i finansijske opravdanosti usluga u sistemima obnovljivih izvora energije [5].

Svi ETU softverski proizvodi su napravljeni da budu: laki za upotrebu, pouzdani, pristupačni i ažurirani sa najnovijom tehnologijom.

ETU softverski proizvodi se mogu koristiti kao pojedinačna rešenja ili se mogu međusobno povezati: isti model zgrade može se koristiti za višestruke inženjerske/simulacione zadatke u celom procesu gradnje objekta, od običnih kalkulacija vrednosti do toplotne procene gubitaka ili hlađenja, dimenzionisanje HVAC opreme, projektovanje cevovodne mreže, hidrauličko balansiranje i dinamička simulacija izgradnje koja uključuje obnovljive izvore energije. ETU softverski alati se mogu koristiti za novogradnju i retro-fit objekte [5].

2.5. Solkane softver

Solkane je softverski paket za izračunavanje termofizičkih svojstava fluida razvijenih od strane nemačke kompanije Solkane, koja prodaje rashladna sredstva. Ovaj softverski paket uključuje i samostalnu aplikaciju za izvršavanje proračuna svojstava za različite rashladne fluide, kao i kompletne proračune ciklusa hlađenja sa kartama rashladnog sredstva, u kojima se mogu nacrtati ciklusi [6].

2.6. TRNSYS softver

Transient System Simulation Program (TRNSYS) je simulacioni program za energetske sisteme. TRNSYS je komercijalni softverski paket koji je razvijen na Univerzitetu Wisconsin. Jedna od prvobitnih primena je bila da se izvrši dinamička simulacija ponašanja solarnog sistema tople vode za tipičnu meteorološku godinu, kako bi se mogla utvrditi dugoročna ušteda takvog sistema [7]. Mnogi različiti energetske sistemi mogu se simulirati u TRNSYS-u - sistemi za solarnu energiju, ventilacija i sistemi za grejanje vode u kućama i sistemi za kogeneraciju. Biblioteke komponenti sadrže komponente koje se koriste u različitim energetske sistemima. Dodatne biblioteke komponenti mogu se kupiti zajedno sa TRNSYS-om. Najveća karakteristika ovog softvera su modularne analitičke metode [7].

3. UPOREDNA ANALIZA SOFTVERA

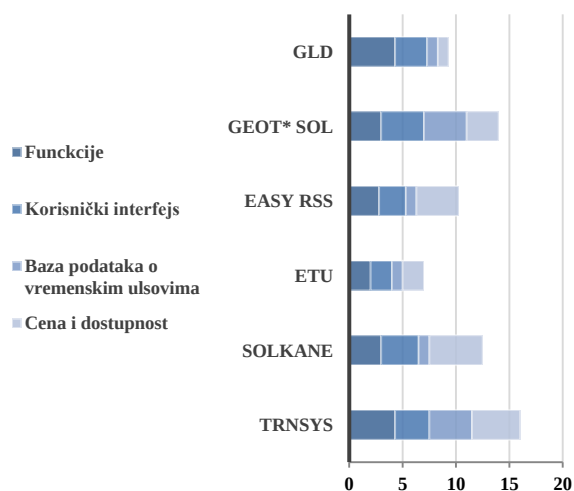
Softveri koji su navedeni u prethodnom poglavlju međusobno su analizirani i ocenjeni prema nekoliko, ovde postavljenih kriterijuma:

1. funkcionalnost softvera,
2. korisnički interfejs,
3. baza podataka o vremenskim uslovima,
4. cena i dostupnost softvera.

Poređenje između softvera je urađeno kako bi se utvrdilo koji od ovih softvera je najbolji za simulaciju i odabir parametara toplotnih pumpi. Bitno je naglasiti da su u ovom radu korišćene probne verzije softvera. Takođe, ocenjivanje je izvršeno na bazi subjektivnog iskustva autora prilikom nabavke, instalacije, probnog rada i primene ovih softvera.

Ukupna ocena svih softvera po navedenim kriterijumima je predstavljena na slici 2. Sa slike može se videti da je najbolje ocenjen softver TRNSYS, koji je ukupno dobio 16 od 20 mogućih poena.

U nastavku rada ovaj softver će biti iskorišćen za simulaciju integracije toplotne pumpe u sisteme za skladištenje tople i hladne vode i za simulaciju različitih kombinacija performansi toplotne pumpe voda-voda.



Slika 2. Uporedna analiza predstavljenih softverskih alata.

4. PRIMERI PRIMENE ODABRANOG SOFTVERA

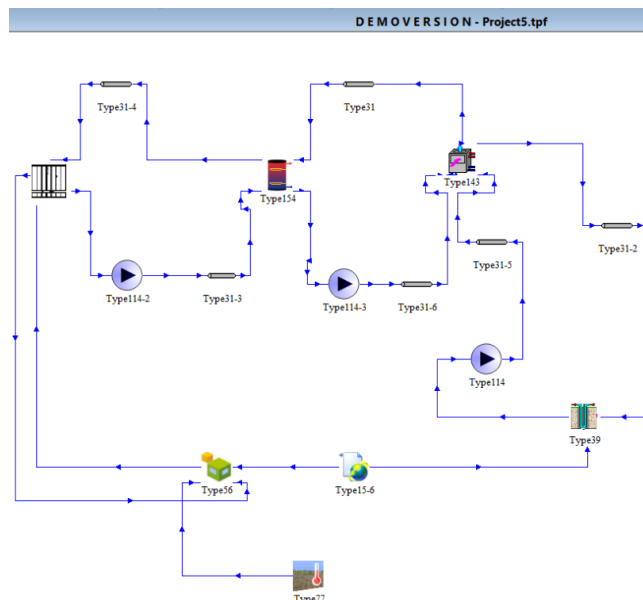
Softver sa najboljim ocenama je TRNSYS softver. Uz pomoć ovog softvera određena je simulacija različitih kombinacija performansi toplotne pumpe tipa voda-voda za porodičnu kuću u Novom Kneževcu.

4.1. Simulacija različitih kombinacija performansi toplotne pumpe tipa voda-voda za kuću u Novom Kneževcu

Objekat se nalazi u Novom Kneževcu i predstavlja stambenu kuću na sprat od oko 120 m². Kuća se greje na prirodni gas i prosečna mesečna potrošnja gasa za zimski period od 6 meseci je oko 350 m³. Grejanje u ovom prostoru vrši se preko radijatora.

Kada se posmatra na duži period, ovaj vid grejanja uopšte nije isplativ. Iz tog razloga je u ovom primeru primenjena simulacija različitih kombinacija performansi toplotne pumpe, čiji rezultat je sezonski grejni faktor (SPF) toplotne pumpe tipa voda-voda, ne bi li se smanjili troškovi grejanja.

Na sledećoj slici 3. prikazana je šema razvijenog modela u simulacijskom studiju u TRNSYS-u.



Slika 3. Šematski prikaz razvijenog modela u TRNSYS-u

Za modelovanje termalnog rezervoara, koji se smatra srcem razvijenog modela u TRNSYS-u, korišćen je vertikalni cilindrični termalni rezervoar. Definisana su dva priključka za povezivanje toplotne pumpe tipa voda-voda i sistem za distribuciju grejanja prostora (radijator) na termalni rezervoar. Čista voda se smatrala radnim fluidom.

Za modelovanje toplotne pumpe u TRNSYS-u, korišćena je jednofazna toplotna pumpa tipa voda-voda, koja je uzeta iz biblioteke (TYPE 668). Toplotna pumpa je modelovana tako da veličina odgovara nešto većoj od 70% maksimalne potrebe za grejanjem kuće. Stambeni objekat je građen 1991. godine, tako da je za specifično toplotno opterećenje uzeto 104 W/m². Iz sledećeg izraza (1) je dobijena ukupna potrebna energija za grejanje kuće.

$$P_e = q * A \quad (1)$$

Gde je q specifično toplotno opterećenje objekta, a A je ukupna neto površina stambenog objekta. Dobijeno je da je ukupna potrebna energija za grejanje ove kuće 12,48 kW.

U sledećoj tabeli 1. prikazane su performanse toplotne pumpe tipa voda-voda.

Tabela 1. Performanse toplotne pumpe

Temperatura ulaznog izvora [°C]	Temperatura ulaznog opterećenja [°C]	Kapacitet grejanja toplotne pumpe [kW]	Ulazna snaga toplotne pumpe [kW]
-10	25	8,195	2,036
-10	35	8,173	2,58
-10	45	8,228	3,266
-10	55	8,359	4,12
-5	25	9,4	2,085
-5	35	9,279	2,608
-5	45	9,203	3,257

-5	55	9,166	4,07
0	25	10,757	2,147
0	35	10,538	2,665
0	45	10,335	3,291
0	55	10,124	4,064
5	25	12,265	2,216
5	35	11,946	2,739
5	45	11,618	3,353
5	55	11,229	4,096
10	25	13,921	2,284
10	35	13,501	2,823
10	45	13,044	3,436
10	55	12,479	4,16
15	25	15,714	2,344
15	35	15,195	2,913
15	45	14,602	3,534
15	55	13,858	4,249

Za modelovanje izvora toplote (voda) u TRNSYS-u, korišćen je vertikalni izmenjivač toplote U-tipa. Radni fluid dobija toplotu iz dubine zemlje u U-tipu izmenjivača toplote i pumpa se u isparivač toplotne pumpe gde se vrši izmena toplote između radnog fluida i rashladnog sredstva toplotne pumpe.

4.2. Rezultati simulacije

Kao rezultat ove simulacije dobijena prosečna vrednost koeficijenta efikasnosti (COP), oko 5,4. U tabeli 2. prikazani su ostali rezultati simulacije.

Tabela 2. Rezultati simulacije

Veličina toplotne pumpe [kW]	11,62
Godišnja energija dobijena toplotnom pumpom [kWh/god]	27 250
SPF [kWh/god]	2,7

Ukupna potrebna električna energija za rad toplotne pumpe tokom cele godine je dobijena iz sledećeg izraza (2):

$$P_{god\ el.en.} = \frac{P_{god}}{COP} = \frac{27\ 250}{5,4} = 5\ 046,3\ Kwh \quad (2)$$

Na osnovu dobijenih rezultata iz simulacije može se zaključiti da je ulaganje u toplotnu pumpu isplativo na dugoročni period. Ako se uzme u obzir da je cena prirodnog gasa veoma visoka, toplotne pumpe tipa voda-voda predstavljaju najekonomičniji vid grejanja, iako su početna ulaganja dosta veća.

Ekonomičnost toplotnih pumpi ogleda se se, pre svega, u što većem koeficijentu grejanja koji predstavlja odnos količine toplotne energije, koju isporučuje toplotna pumpa i količine električne energije potrebne za rad toplotne pumpe. Od svih alternativnih grejanja najveću perspektivu imaju toplotne pumpe (tipa voda – voda), jer su se pokazale kao ekonomični sistemi za grejanje u dobro izolovanim kućama.

5. ZAKLJUČAK

Primena obnovljivih izvora energije i njihovo eksploataisanje je od velike važnosti i predstavlja bitan segment energetske-ekološke strategije razvijenih zemalja. Najefikasnija primena geotermalnih toplotnih pumpi je za zagrevanje i hlađenje. Ovakav vid korišćenja njenih potencijala može se primeniti kako za grejanje tako i hlađenje stambenih, industrijskih i drugih tipova objekata. Ono što je najbitnije kod ovakvih sistema jeste njegova energetska efikasnost koja može iznositi i 70% u odnosu na druge sisteme grejanja. Ako se uzme u obzir da se ovom investicijom rešava grejanje i da postoji mogućnost hlađenja prostora onda je ušteda pri radu toplotne pumpe znatno veća.

6. LITERATURA

- [1] Dragana Gašić, „Analiza primjene toplotne pumpe za grijanje i klimatizaciju poslovnog prostora”, Diplomski rad, Fakultet tehničkih nauka, Novi Sad, 2018.
- [2] www.groundloopdesign.com/ (pristupeljeno u junu 2019.)
- [3] www.valentin-softwar.com/ (pristupeljeno u junu 2019.)
- [4] www.easy-rss.co.uk (pristupeljeno u junu 2019.)
- [5] www.etu-software.com (pristupeljeno u junu 2019.)
- [6] M. Sikora, “Parameters selection of a heat pump using the Solkane software”, Koszalin University of Technology (KTU), Poljska, 2017
- [7] www.trnsys.com/demo/ (pristupeljeno u junu 2019.)
- [8] S.M. Al Zahrani, F.L Tan, F.H. Choo, “A TRNSYS Simulation Case Study on Utilization of Heat Pump For both Heating and Coolin”, Nanyang Technological University, 2012
- [9] H.K. Bakhtiari, “Developing a dynamic model of ground-source heat pump system to evaluate different components size”, Department of Energy Technology, SE-100 44, Stokholm, Švedska

Kratka biografija:



Anja Subotin je rođena u Novom Kneževcu 1993. god. Diplomirala je na Fakultetu tehničkih nauka 2016. god. sa temom "Idejno rešenje napajanja kuće u Novom Kneževcu iz obnovljivih izvora energije". Iste godine upisuje Master akademske studijena Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva-Elektroenergetski sistemi. Master rad je odbranila 2019.god.



Vladimir Katić je rođen 1954. god. u Novom Sadu. Doktorirao je na Univerzitetu u Beogradu 1991. god. Od 2002. god. 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.

**MODEL DDOS NAPADA APLIKACIONOG SLOJA GAMING SERVERA
EKSPLOATACIJOM KLIJENTSKE APLIKACIJE****MODEL OF APPLICATION LAYER BASED DDOS ATTACK ON A GAMING SERVER
BY EXPLOITING THE CLIENT APPLICATION**

Nikola Gavrić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu se analiziraju problemi i posledice DDoS napada na aplikacionom sloju gaming servera. Postavljena je hipoteza, da deo tih napada nastaje eksploatacijom klijentske aplikacije. Radi potvrde postavljene hipoteze, izvršena je simulacija napada u laboratorijskim uslovima. Dobijeni su rezultati u formi logova na serveru koji su prikazani u grafičkoj formi. Njihovom evaluacijom došlo se do određenih zaključaka na osnovu kojih je predloženo je rešenje za zaštitu od ovih napada, koje se bazira na primeni machine learning tehnike na mrežnim firewall-ima.

Ključne reči: DDoS, protocol-specific DDoS, DDoS u gaming industriji, firewall, machine learning

Abstract – This paper analyzes problems and consequences of application layer based DDoS attacks on gaming servers. The hypothesis is that some of these attacks are caused by exploitation of the client application. A simulation was performed in a laboratory in order to prove these assumptions. The obtained logs from the server are plotted. There are several conclusions based on the data gathered through the simulation process and there's a suggested solution relying on machine-learning based network firewalls.

Keywords: DDoS, protocol-specific DDoS, DDoS in gaming industry, firewall, machine learning.

1. UVOD

Na samom početku računarstva, pojam bezbednosti se uglavnom vezivao za zaštitu podataka na izolovanim računarima. Nastankom računarskih mreža i njihovim permanentnim širenjem i razvojem, naročito sa aspekta primene novih, naprednijih tehnologija i servisa, pojavili su se brojni i veoma sofisticirani izazovi u oblasti zaštite podataka koji se razmenjuju različitim komunikacionim kanalima. Motivi za napade su razni, od hakerskog aktivizma do iznude i potiskivanja konkurencije. Napadi su u početku bili usmereni ka krajnjim korisnicima, sa ciljem da im se ubacivanjem malicioznog softvera degradiraju performanse računara i ukradu važni podaci (passwordi, brojevi kreditnih kartica i sl.). Drugim rečima, ubacivanjem malicioznog softvera preuzima se

kontrola nad korisničkim računarima, radi izvođenja koordinisanih napada. S druge strane, napadi usmereni ka velikim sistemima realizuju se najčešće u formi DDoS (eng. *Distributed Denial of Service*) napada, čiji je osnovni cilj da izazovu prekid korišćenja određenih servisa i naruše performanse mreže, a da se pri tome sakrije identitet napadača.

Sa promenama u društvu, menjali su se motivi, ali i metode koje su napadači koristili u kreiranju DDoS napada, ali je cilj ostao isti, da se korisnicima onemogući ili oteža pristup određenim servisima ili resursima i da se sakrije identitet. Prvobitni napadi ovog tipa bili su DoS (eng. *Denial of Service*) napadi i bili su usmereni na mreže, sisteme i određene servise.

Generalno posmatrano, DDoS napadi se mogu izvesti na bilo kojem sloju mrežne komunikacije, ali zavisno od toga na šta su usmereni napadi i kako se izvode mogu podeliti u sledeće tri kategorije:

1. Napadi na propusni opseg (eng. *bandwidth*) - usmereni su na preopterećenje tj. narušavanje performansi mrežnih resursa u pogledu protoka ili propusne moći uređaja, a u slučaju napada velikog intenziteta mogu dovesti do zagušenja i prekida linka između provajdera i korisnika (kod zagušenja dolazi i do usporavanja korisničkog saobraćaja, pojave retransmisija koje proizvode dodatni saobraćaj i kašnjenja).
2. Protokolski napadi - izvode se preko tačno određenih protokola, koristeći njihove nesavršenosti.
3. Aplikacioni napadi - ciljaju softverske nedostatke aplikacija na kojima se baziraju korisnički servisi, kao npr. nedostatke web servera.

Na aplikacionom sloju, DDoS napadi se mogu podeliti u tri grupe [1]:

1. Napadi koji se izvode pravom „poplavom“ novih sesija na resursima u mreži (eng. *Session flooding*) - izvodljivi su isključivo na jednostavnim aplikacijama koje ne podrazumevaju verifikaciju i autorizaciju korisnika (npr. veb sajtovi) i nisu predmet ovog rada, iz razloga što to nije svojstveno za gaming servere.
2. Poplava i pravo bombardovanje novim zahtevima (eng. *Request flooding*)
3. Asimetrični napadi – nastaju kada zahtev za izvršavanje neke operacije koristi znatno manje resursa na klijentskoj strani, naspram same operacije koja se izvršava na serveru (ovi napadi

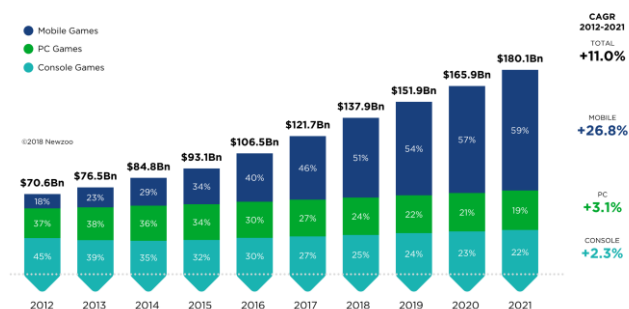
NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio doc. dr Živko Bojović.

su takve prirode da iziskuju znatne hardverske resurse servera).

2. DDoS NAPADI U INDUSTRIJI IGRICA

Većina napada na aplikacionom sloju cilja HTTP, pa je s toga *cybersecurity* industrija isključivo usmerena ka zaštiti web sajtova i web servisa. Ovaj rad se fokusira na aplikacioni sloj, specifično na *gaming* aplikacije, jer *gaming* industrija trenutno beleži velik porast i rapidnu ekspanziju. Biće demonstrirane potpuno nove metode DDoS napada koje proizilaze iz eksploatacije klijentskih aplikacija i metodi zaštite od istih, sa osvrtom na postojeće mehanizme i njihove nedostatke.



Slika 1. Rast gaming industrije [2]

Potrebno je naglasiti da je većina industrije igrice orijentisana ka tzv. *online gaming*-u. U ovom radu pažnja je usmerena prvenstveno ka *online gaming*-u na PC platformi, s time što su koncepti koji će biti prikazani, uz male modifikacije primenljivi na mobilnoj i konzolnoj platformi.

Aplikacije na *gaming* serverima su takve prirode da zahtevaju znatne hardverske i mrežne resurse. Tome doprinose novi trendovi, na primer da se dozvoljava pokretanje aplikacije bez posedovanja potpunog sadržaja iste, a da se zatim tokom korišćenja aplikacije ostatak podataka (ili čak samo neophodni podaci) preuzima sa nekog servera. Još jedan bitan faktor koji utiče na upotrebu resursa servera jesu dozvoljene modifikacije klijentske aplikacije (eng. *Mods/add-ons*), koje mogu da predstavljaju dodatni stepen slobode za napadače.

2.1. Eksploatacija klijentske aplikacije

Kada se govori o bezbednosti klijentske aplikacije, najčešće se misli na zaštitu od napada u smislu prevencije mogućnosti da korisnik aplikacije na nedozvoljen način ostvari prednost u odnosu na ostale korisnike ili drugu vrstu dobiti.

Glavni cilj pri kreiranju aplikacije jeste njena funkcionalnost, a zatim sve ostalo. Jedan od glavnih uzroka ovakve strategije jeste činjenica da mnoge kompanije teže da aplikacije puste u prodaju pre krajnjeg roka (eng. *Deadline*) po svaku cenu, jer to igra veliku ulogu pri inicijalnoj prodaji. Ovakav pristup ne ostavlja dovoljno vremena za intenzivno testiranje i svodi se na popravke *bag*-ova i poboljšanje aplikacije u njenim kasnijim revizijama. Primenjeni odbrambeni mehanizmi u *online gaming* aplikacijama se mogu podeliti na:

1. Potpunu zaštitu na serverskoj strani (eng. *Server-sided checking/protection*)
2. Distribuiranu zaštitu

Potpuna zaštita na serverskoj strani podrazumeva da se kompletno procesiranje vrši na serverskoj aplikaciji. Ovakav pristup obezbeđuje bolju zaštitu od napada, ali po cenu veće hardverske zahtevnosti i većeg kašnjenja u procesiranju zbog provera dolazećih paketa, koji moraju biti u skladu sa pravilima. Suštinska prednost ovog metoda naspram distribuiranog je u tome što napadačima smanjuje stepen slobode.

Pod pojmom distribuirana zaštita se podrazumeva sistem kod koga se deo procesiranja vrši na klijentskoj strani (određene varijable koje se koriste u procesiranju prisutne su samo na klijentskoj aplikaciji), dok je uloga servera da koordinira rad klijenata, obavlja deo procesiranja, vrši određene provere i dr.

2.2. Trenutno stanje u oblasti zaštite od DDoS napada

Problem zaštite od DDoS napada predstavlja danas jedan od gorućih problema u *gaming* industriji i šire. Metode kojima se pokušava obezbediti zaštita od DDoS napada se mogu podeliti u dve osnovne grupe (faze):

1. Prevencija DDoS napada
2. Ublažavanje DDoS napada (eng. *mitigation*) koji su u toku.

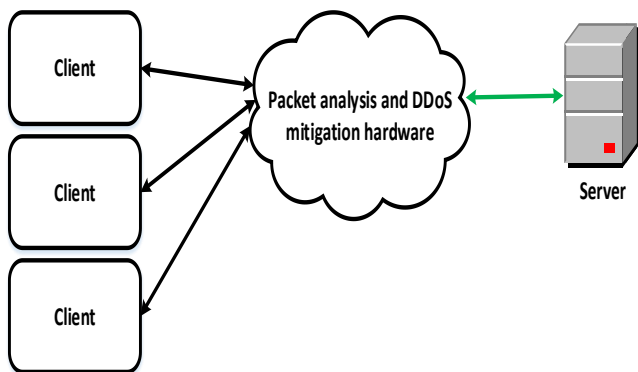
Prilikom dizajniranja aplikacije, odbrambeni mehanizmi nisu u prvom planu i postoje tendencije da se ceo *cybersecurity* domen prepusti provajderu takve usluge (npr. *Cloudflare*). Ovakva rešenja se nazivaju eksternim, ona su najjeftinija i najbrža, ali sa sobom povlače i određene nedostatke.

Najveći nedostatak ovakvih rešenja jeste to što su redno izvedena (slika 2). Naime, radi jednostavnosti implementacije, izdvoje se određeni resursi na cloudu koji se potom ponašaju kao *firewall* i *proxy* server za komunikaciju sa klijentima.

Loša strana redne implementacije jeste u dodatnom kašnjenju koje prouzrokuje fizička ili bolje rečeno geografska udaljenost između *game* servera i *cybersecurity* servera provajdera. Potrebno je istaći da dodatno kašnjenje mogu da unesu algoritmi za detekciju napada i algoritmi za druge vidove zaštite koji se ne odnose na DDoS. U HTTPu ovo nije veliki problem jer navedena kašnjenja gotovo da ne utiču na QoS i QoE.

Međutim, posmatrano sa aspekta *gaming*-a se najčešće (zbog *real-time* prirode) zahtevaju što manje vrednosti kašnjenja. Neke prihvatljive vrednosti kašnjenja su do 50ms, pa je svaka eliminacija kašnjenja dragocena. Rešenja ovog problema su uglavnom privatna i postoje razne implementacije koje su prilagođene specifičnim potrebama nekog preduzeća.

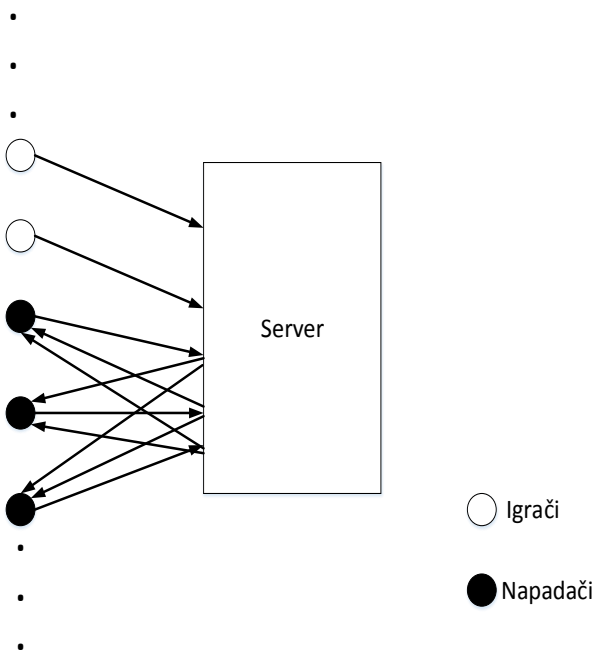
Izazovan deo u borbi protiv DDoS napada na aplikacionom jeste detekcija, s obzirom na to da se napadi lako mogu stopiti u regularan saobraćaj, pritom poštujući sva pravila bezbednosnih mehanizama.



Slika 2. Redno izvedena DDoS zaštita

3. SIMULACIJA

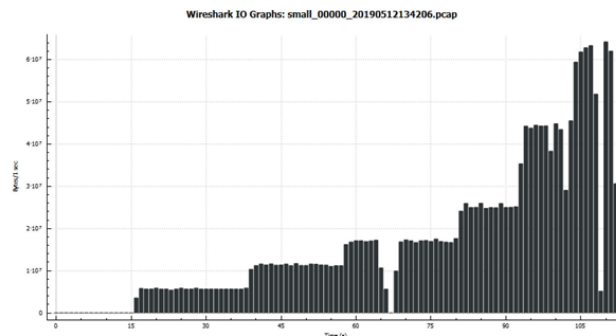
Radi potvrđivanja pretpostavki vezanih za DDoS napad na aplikacionom sloju, izvršena je simulacija u laboratoriji sa 30 računara na Računarskom fakultetu u Beogradu. Relevantne specifikacije računara su: osmojezgrani procesor AMD FX-8350, 16GB DDR3 RAM, Microsoft Windows 10 OS i svi su povezani na 1GB/s ethernet LAN (*Local Area Network*). Za demonstraciju napada specifičnog za aplikacioni protokol (eng. *Protocol specific*) korišćena je igra *World of Warcraft* (verzija 12340) kao klijentska aplikacija, a serverska aplikacija je emulator dostupan kao open source kod pod nazivom Trinity Core [3]. Napad se sastojao iz slanja velike količine poruka (paketa), putem kanala koji su rezervisani za razmenu informacija između *Add-ona* (*Addon Channels*). *Add-ons* su softverske ekstenzije klijentske aplikacije koje su najčešće open source i prave se isključivo iz razloga što omogućavaju dodatne funkcionalnosti koje se mogu isprogramirati. Za napad je korišćena API funkcija *SendAddonMessage* [4].



Slika 3. Šematski prikaz DDoS napada tokom simulacije

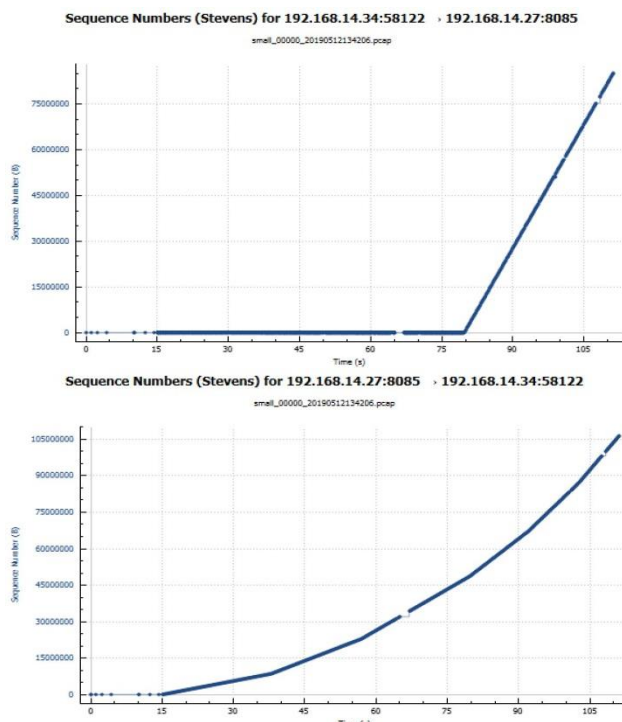
3.1. Efekti napada

Izmerene vrednosti potvrđuju hipotezu da je *protocol-specific* napad na aplikacionom sloju ozbiljna pretnja po bezbednost servera. Na slici 4. može se videti da server ne poseduje adekvatan mehanizam zaštite i da je ukupan protok u vrlo kratkom roku dostigao maksimum linka.



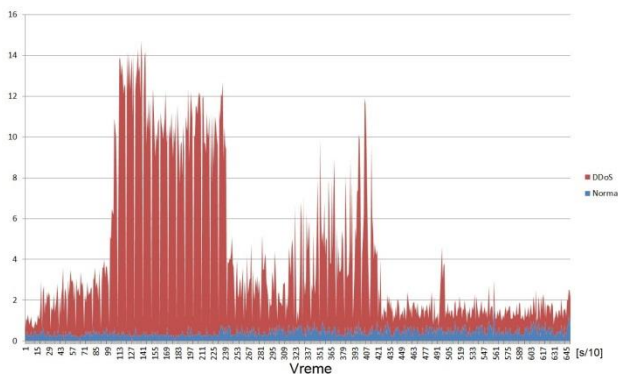
Slika 4. Protok na serveru pre (do 15s) i tokom DDoS napada

Intuitivniji uvid u broj poslatih i obrađenih paketa se može dobiti praćenjem rednih brojeva paketa, što je ekvivalentno njihovoj kumulativnoj sumi kao što je prikazano na slikama 5a i 5b.



Slika 5. Broj paketa između jednog od napadača i servera (a) i ukupna količina paketa na serveru (b)

Na slici 6. je prikazana iskorišćenost procesora u *Kernel-space*, što predstavlja sistemske pozive na linuxu inicirane većinski od strane programa koji se nalaze u *User-space*. Iz *Kernel-space*-a se upravlja mrežnom karticom i ostalim ulazno izlaznim operacijama (npr. poput pisanja na HDD (Hard Disk Drive) ili RAM). Važno je naglasiti, da se čak i sam *scheduler* (proces koji bira naredni proces za obradu od strane procesora) nalazi u *Kernel-space*.



Slika 6. Opterećenost procesora u *Kernel-Space*

4. ZAKLJUČAK

Rezultati jasno ilustruju nemogućnost nezaštićenog servera da se nosi sa ovakvom vrstom napada. Prediktivnom analizom možemo zaključiti da standardni metodi odbrane poput ograničenja protoka ili zaštite na nižim slojevima TCP/IP stack-a ne mogu da zaštite server.

Iz rezultata se vidi da je količina generisanog saobraćaja linearno proporcionalna količini primljenog saobraćaja. Tj za svaki poziv *SendAddonMessage* usmeren ka grupi korisnika, poruka koju u vidu argumenta ove funkcije pošalje napadač se retransmituje ka svima u navedenoj grupi. Dakle ako je x broj korisnika koji primaju poruke poslate od strane jednog napadača i ako je ukupan broj poslatih malicioznih poruka y opisan funkcijom $f: \mathbb{N} \rightarrow \mathbb{N}$, tada u slučaju jednog napadača važi sledeće:

$$y = f(x) = x + 1 + c \quad (1)$$

Izdvojena poruka plus 1 se pojavljuje kao poruka koja se inicijalno šalje serveru, dok x označava broj poslatih poruka od strane servera. Faktor c označava dodatne poruke prouzrokovane ovim saobraćajem, poput potvrde prijema itd.

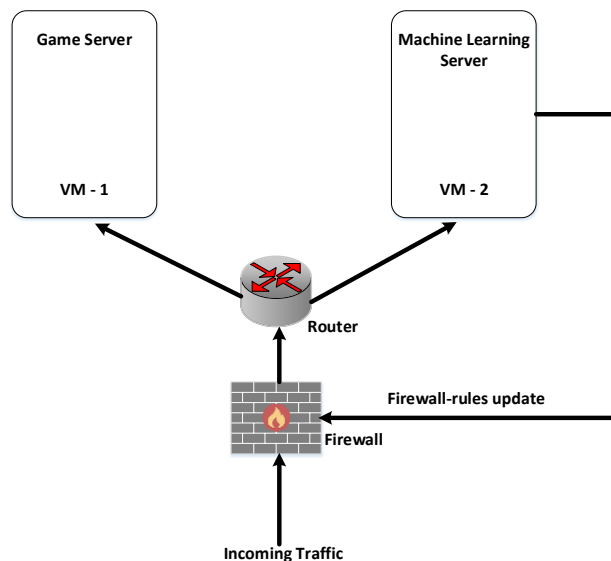
Ukoliko dodamo više napadača kao tokom simulacije napada, u situaciji kada su svi primaoci malicioznih poruka takođe i predajnici istih i definišemo funkciju $g: \mathbb{N} \rightarrow \mathbb{N}$ koja opisuje ukupan broj poslatih poruka između servera i napadača, onda je:

$$y = g(x) = x * f(x) = x * (x + 1) + cx \quad (2)$$

Drugim rečima, ukupan broj poslatih poruka raste srazmerno kvadratu broja napadača. Iako su napadi bili različitih intenziteta, ovakav zaključak je i dalje evidentan na slici 5b. Intuitivniji opis jeste da zbog konstantne brzine slanja, kumulativna suma poruka raste linearno tokom vremena. Međutim kada tokom vremena dodajemo još napadača, jer nismo bili u mogućnosti da pokrenemo više napada istovremeno, dobijemo kumulativnu funkciju koja je nelinearna tokom vremena, a čije restrikcije su linearne funkcije kojima tokom vremena raste faktor skaliranja (Nagib). Uopšteno to predstavlja dobru ilustraciju kvadratne zavisnosti količine poslatih poruka od broja napadača.

Rešenje za ovakvu vrstu DDoS napada nazire se u machine-learning klasifikatorima. Predloženi mehanizam zaštite (slika 7) bi, uz uvođenje metrike ξ kojom bi se opisala „Cena” poziva API funkcija, bio u stanju da zaštiti server od navedenih DDoS napada. Problemi koji se

javljaju pri upotrebi ovakvog modela su vezani isključivo za pristrasnost ka aplikaciji kao i činjenica da se oslanja na dobru procenu parametra ξ .



Slika 7. Predloženi mehanizam zaštite servera

mogućnosti za pravljenje modela zaštite su gotovo neograničene, ali je neophodno uzeti u obzir realna ograničenja, pogotovo kompleksnost detektora, a samim tim i brzinu detekcije koja se izuzev hardverskih i softverskih ograničenja svodi na *Uncertainty principle*. Drugim rečima detektor nema neograničeno vreme da donese odluku o detekciji, ali istovremeno treba da prikupi dovoljnu količinu informacija, što iziskuje vreme. Detektor takođe treba da demotivise napadače, time što će ih naterati da traže još kompleksnije načine za napad. Iz prethodno navedenih razloga, problem detekcije DDoS napada na aplikacionom sloju u *gaming* industriji ostaje izazovna oblast, gde je cilj napraviti izvodljiv model zaštite koji će biti u stanju da u potpunosti ili makar u velikoj meri onemogući manifestaciju štetnih efekata napada, a da pritom ne naruši QoS i QoE regularnih igrača.

5. LITERATURA

- [1] Gaurav Somani, „DDoS attacks in cloud computing: Issues, taxonomy, and future directions,” *Computer Communications*, 107 30-48, 2017.
- [2] <https://newzoo.com/insights/articles/global-games-market-reaches-137-9-billion-in-2018-mobile-games-take-half/>
- [3] <https://github.com/TrinityCore/TrinityCore>
- [4] https://wow.gamepedia.com/API_C_ChathInfo.SendAddonMessage

Kratka biografija:



Nikola Gavrić rođen je u Novom Sadu 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Telekomunikacioni sistemi odbranio je 2019.god.
kontakt: nikolagavric021@gmail.com

VELIKI PODACI U ELEKTROENERGETSKIM SISTEMIMA**BIG DATA IN ELECTRIC POWER SYSTEMS**Stefan Đorđević, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – U radu su predstavljeni tehnologija velikih podataka i načini primene velikih podataka u elektroenergetskim sistemima. Nakon predstavljanja koncepta veliki podataka, tehnika za prikupljanje, obradu i skladištenje podataka, i metoda za analizu podataka, predloženi su primeri primene velikih podataka u mnogobrojnim procesima električnih mreža, kao i uticaja na relejnu zaštitu.

Abstract – This paper presents the technology of big data and its applications in electric power systems. After presenting the concept of big data, techniques for data collection, processing and storage and methods for data analysis, examples of the application of big data in many electrical grid processes, as well as impacts on relay protection, are proposed.

Ključne reči: Veliki podaci, Elektroenergetski sistemi, Relejna zaštita, Pametne mreže.

1. UVOD

Veliki podaci (engl. *Big data*) označavaju velike i kompleksne skupove podataka kod kojih tradicionalne aplikacije za obradu podataka nisu primenljive. U svrhu otkrivanja znanja, ovi podaci moraju biti prikupljeni, skladišteni i analizirani. Primenom odgovarajućih analitičkih metoda, iz podataka se mogu izvući vredne informacije na osnovu kojih se mogu doneti korisne odluke. Donesene odluke se mogu iskoristiti za proaktivno delovanje u različitim procesima mnogih industrija [1]. U elektroenergetskom sektoru, uz primenu koncepta pametnih mreža i sve većeg prodora informacionih i komunikacionih tehnologija, digitalizacija je stvarnost. Paralelno s tim, elektroenergetski sistemi se suočavaju sa različitim izazovima koji se odnose na operativne aspekte, kao što su potrebe za smanjenjem operativnih troškova i povećanjem efikasnosti, povećanjem udela obnovljivih izvora energije, električnih automobila i povećanjem sigurnosti i pouzdanosti same mreže. Ovi izazovi, zajedno sa pojavom senzora, naprednih uređaja za merenje, uređaja za komunikaciju i inovacijama koje donose veliki podaci, predstavljaju nove mogućnosti, posebno zato što će sledeća generacija energetske sistema – pametne mreže, biti izuzetno intenzivna za podatke [2]. Vredne informacije i komunikacione tehnologije generišu veliku količinu podataka u različitim vremenskim okvirima. Tako da trenutna transformacija elektroprivrede u sistem sa više varijabilnih i distribuiranih resursa zahteva nov način

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je dr Duško Bekut, red.prof.

prikupljanja i tumačenja tih podataka kao osnovni zahtev za sprovođenje analiza elektroenergetskog sistema. Upravo, cilj ovog rada je da predstavi ulogu tehnologije velikih podataka u elektroenergetskim procesima i opiše kroz primere primene vrednost koju ona može doneti.

U drugom delu rada predstavljen je koncept i uloga velikih podataka u elektroenergetici, u trećem analiza, a u četvrtom delu primena velikih podataka. Peti deo je zaključak, a u šestom literatura.

2. KONCEPT I ULOGA VELIKIH PODATAKA U ELEKTROENERGETICI

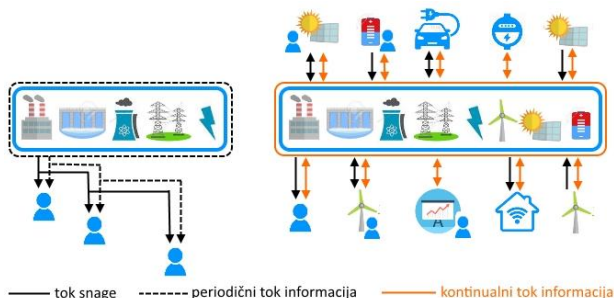
Veliki podaci su apstraktni koncept. Ne postoji jedna sveobuhvatna definicija za velike podatke, jer ona varira zavisno od toga ko ih opisuje i u kom kontekstu. Uopšteno, veliki podaci označavaju skupove podataka koji se ne mogu percipirati, prikupiti, upravljati i obrađivati tradicionalnom informacionom tehnologijom i softverskim/hardverskim alatima unutar prihvatljivog vremenskog okvira.

Tehnologija velikih podataka mora da obuhvata tehnike za trenutnu, ili skoro trenutnu, analitiku podataka i vizuelizaciju, kao i napredne tehnologije za generisanje i prikupljanje, prenos i skladištenje podataka, kako bi se stvorile korisne informacije i na osnovu njih sprovodile proaktivne akcije. Takođe, mora i da obezbedi takvu platformu koja ima sposobnost da automatizuje ceo proces [3].

Tokom protekle i ove decenije, elektroprivreda prolazi kroz fundamentalne promene. Prvi veliki uzrok je drastična promena portfolija proizvodnje kroz povećan udeo obnovljivih izvora energije. Pouzdanost i sigurnost mreže će morati biti pažljivo osmišljeni za takve promene. Drugi glavni uzrok je povećanje nivoa korišćenja distributivne mreže u odnosu na prenosnu mrežu na velikoprodajnom nivou [4]. Ovo pokreće osnovna pitanja za korišćenje i rad sistema u budućnosti. U ovom izazovu, podaci će verovatno igrati važnu ulogu. Vrednost podataka verovatno će energetske industriji pružiti obnovljenu paradigmatu koja vodi renesansi njenih poslovnih modela.

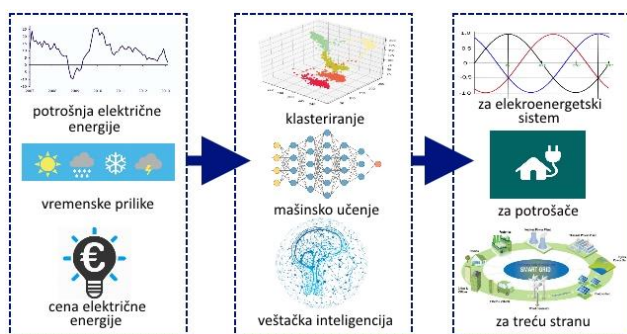
Pametna mreža se može posmatrati kao ogromna senzorska mreža, koja stalno prikuplja podatke iz različitih izvora, kao što su raznovrsni senzori, pametna brojlara, pametni uređaji, električna vozila ili čak meteorološke stanice. Zatim, ova ogromna količina podataka može se transformisati u uvidne radnje primenom upravljanja velikim podacima i napredne analitike. Ovi uvidi mogu pomoći u poboljšanju

efikasnosti, pouzdanosti i održivosti elektroenergetske mreže kroz različite slučajeve upotrebe, kao što su otkrivanje i obnavljanje prekida, preventivno održavanje, oblikovanje obrazaca potrošnje potrošača, automatizacija distributivnog sistema i kontrola emisije. Na slici 2.1. prikazana je struktura tradicionalnih i pametnih mreža. Tradicionalna električna mreža uključuje jednosmerni prenos, što znači da snaga teče od generatora do potrošača. Pametni mrežni sistemi, s druge strane, uključuju dvosmerni prenos podataka, sistem koga pokreću podaci i obnovljivi izvori energije za pružanje dodatnih usluga potrošačima, distributerima i snabdevačima.



Slika 2.1. – Tradicionalna mreža naspram pametne mreže

Slika 2.2. prikazuje obrazac primene analitike velikih podataka u aplikacijama elektroenergetskog sistema. Proces prikupljanja podataka počinje sa različitim izvorima kao što su pametna brojlara, energetska tržišta i vremenski senzori. Stručnjaci tada mogu primeniti napredne statistike i tehnike mašinskog učenja, kao što su klasteriranje i duboko učenje, kako bi stvorili uvid u podatke koji može biti delotvoran [5]. Rezultati tih tehnika primenjenih na podacima se mogu koristiti za poboljšanje rada i planiranja mreže, kao i iskustva potrošača ili mogu biti iskorišćeni od strane treće strane u drugim poslovnim svrhama.



Slika 2.2. – Obrazac primene tehnologije velikih podataka u elektroenergetici

Kao i u svakoj drugoj oblasti industrije, za distribuciju električne energije veliki podaci predstavljaju veoma snažan način za poboljšanje efikasnosti. Dok elektroprivrede poseduju veliki broj strukturiranih podataka prikupljenih iz njihovih mrežnih mernih sistema, oni takođe moraju da se bave nestrukturiranim izvorima podataka kao što su mape, fotografije, istorijski podaci i drugo [5]. Pretvaranje ovih podataka u korisniju formu zadatak je tehnika velikih podataka. Scenariji primene velikih elektroenergetskih podataka obuhvataju sve aspekte proizvodnje, prenosa, podstanice, distribucije, korišćenja i raspoređivanja električne energije. Veliki

podaci o električnoj energiji obećavaju perspektivu primene u predviđanju proizvodnje obnovljivih izvora energije, analizi rizika sigurnosti mreže i ranom upozoravanju, planiranju održavanja komponenti, optimalnom snabdevanju, nadzoru i dijagnostici stanja mreže, upravljanju životnim ciklusom opreme, praćenju i proceni kvalitete električne energije, lokaciji niskog napona kontrola distributivne mreže, procena rizika i rano upozorenje preopterećenosti distributivne mreže, upravljanje protiv krađe električne energije, analiza karakteristika opterećenja, predviđanje opterećenja, upravljanje potražnjom, analiza i upravljanje energetsom efikasnošću potrošača i tako dalje [6].

2.1 Karakteristike velikih podataka

Uobičajena definicija velikih podataka uključuje klasifikaciju njenih karakteristika pomoću 5V-a [6]:

- 1) OBIM (engl. *Volume*)
Odnosi se na ogromnu količinu generisanih podataka.
- 2) BRZINA (engl. *Velocity*)
Odnosi se na brzinu prenosa, skladištenja, obrade i analize podataka.
- 3) RAZNOVRSNOST (engl. *Variety*)
Prema obimnim izvorima podataka u pametnoj mreži, formati i dimenzije podataka su različiti po strukturi.
- 4) TAČNOST (engl. *Veracity*)
Odnosi se na potrebu uzimanja u obzir pouzdanost i doslednost korišćenih podataka.
- 5) VREDNOST (engl. *Value*)
Odnosi se na sposobnost da se izvuku vredne informacije iz ogromne količine podataka.

Prethodna svojstva mogu biti dopunjena sa 3E [6]:

- 1) ENERGIJA (engl. *Energy*)
Predstavlja uštedu energije koja se može postići primenom tehnika velikih podataka.
- 2) RAZMENA (engl. *Exchange*)
Predstavlja razmenu informacija između energetske i drugih sektora.
- 3) EMPATIJA (engl. *Empathy*)
Predstavlja pružanje boljih energetske usluga kojima se mogu ispuniti potrebe korisnika.

3. ANALIZA VELIKIH PODATAKA

Glavni cilj upotrebe analize velikih podataka je izvlačenje korisnih informacija (vrednosti) iz podataka. Ova vrednost se može izvući iz prikupljenih podataka nakon obavljanja analitike na podacima kao što je prikazano na slici 3.1. [3] Preduzeća i potrošači mogu donositi odluke na osnovu dobijene vrednosti.



Slika 3.1. – Tok obrade podataka

Kao što je prikazano na slici 3.1. prvi korak obuhvata prikupljanje podataka iz više heterogenih izvora u mreži,

kao i izvan nje. Prikupljeni podaci u izvornom obliku mogu se učitati u prostor za skladištenje (računarstvo u oblaku). Nakon prikupljanja podataka, sledeći korak je predobrada podataka. Ovi podaci se tada čiste da bi se uklonili pogrešni i nepotpuni podaci i zatim transformišu u format pogodan za ciljnu obradu i analizu i učitavaju u skladište podataka. Sada se tehnike analitike podataka primenjuju na prethodno obrađene podatke kako bi se izvukla vrednost na osnovu koje se mogu izvršiti neke informisane akcije ili odluke [3].

Najvažnija faza sistema obrade velikih podataka je analiza podataka, odnosno analitika koja je osnova za otkrivanje dragocenih informacija i podršku donošenja odluka [7].

Rudarenje podataka (engl. *Data Mining*) je proces otkrivanja skrivenih i nepoznatih obrazaca u velikim skupovima podataka, korišćenjem metoda koje su u preseku statistike, mašinskog učenja (kao deo veštačke inteligencije) i sistema baza podataka. Time predstavlja interdisciplinarnu podoblast informatike, statistike i primenjene matematike, čiji je opšti cilj izvlačenje informacija iz masivnih, nepotpunih i nejasnih podataka sa šumom i transformacija tih informacija u razumljivu strukturu radi dalje upotrebe [6,7].

Svrha različitih tipova analitike „kao usluge“ se menja kako se kreće duž kontinuuma vrednosti na sledeći način [3]:

- 1) Deskriptivna analitika ima za cilj da pruži informacije o tome šta se dogodilo
- 2) Dijagnostička analitika ima za cilj razumevanje uzroka događaja
- 3) Prediktivna analitika se koristi kako bi se napravila verovatna predviđanja s ciljem da se utvrdi što bi se moglo dogoditi u budućnosti
- 4) Preskriptivna analitika se primenjuje da bi se identifikovao najbolji ishod događaja

Tehnike koje su najprikladnije za elektroprivredu su:

Učenje pravila asocijacije, Klaster analiza, Faktorska analiza, Korelaciona analiza, Regresiona analiza i Inferencijska statistička analiza.

4. PRIMENE VELIKIH PODATAKA

U nastavku će se na konkretnim primenama razmotriti kako jedan ovakav koncept može naći svrhu u okruženju elektroenergetskog sistema i kakav značaj za isti može predstavljati.

Prognoza proizvodnje distributivnih izvora

Pošto su kapaciteti za proizvodnju električne energije distributivnih resursa (PV i vetroelektrane) usko povezani sa klimatskim uslovima koji karakterišu slučajnost i intermitentnost, izazov je povezati ih sa električnim mrežama. Njima se može upravljati efikasnom analizom velikih podataka koja omogućava precizniju prognozu proizvodnje. Podaci koji omogućavaju prognozu su: temperatura vazduha, oblačnost, prelazak oblaka preko sunca, zračenje sunca, brzina i snaga vetra, smer i pravac vetra, izlazne snage, položaj panela/vetroturbin, istorijska proizvodnja, GIS podaci, satelitski snimci, topografski podaci itd. A

vrednost izvučena iz analize može se primeniti za balansiranje energijom u mreži, efikasno upravljanje tržištom energije kao i efikasno korišćenje energije.

Prognoza potrošnje

Faktori koji utiču na prognoziranje potrošnje mogu se podeliti u dve kategorije:

- 1) tradicionalni faktori – vremenski uslovi, doba dana, godišnje doba, slučajni događaji i poremećaji u mreži,
- 2) faktori pametne mreže – cene električne energije, distribuirani izvori energije, električna vozila, pojedinačna upotreba kućnih aparata, društveni događaji, uređaji za grejanje/hlađenje.

Na osnovu prethodnih podataka, kroz analizu, može se analizirati ponašanje potrošača pri potrošnji električne energije i pružiti raznovrsne i personalizirane usluge i ohrabriti kupce da učestvuju u balansiranju proizvodnje i potrošnje i programima energetske efikasnosti.

Detekcija i predviđanje prekida

Veliki podaci nude načine da se efikasnije reaguje na retke događaje koje izazivaju kvarove. Podaci o vremenskim prilikama u realnom vremenu i istorijski podaci o vremenu, kao i podaci GIS-a, u kombinaciji sa istorijskim podacima o kvarovima i ispadima u određenom delu fidera, podacima o stanju energetske opreme, podacima iz merenja i zaštite (pametna brojila, IED, SCADA, PMU) koji proširuju nadzor i svesnost situacije na fideru mogu se iskoristiti za brzo određivanje uzroka kvara (oluje, grmljavine, poplave, požari, snežni nanosi), ali i za preventivno delovanje i bolju pripremu za nastupajuće poremećaje (npr. isključivanje dela mreže i obezbeđivanje rezervnog napajanja, sprečavajući oštećenje opreme, kolaps u sistemu i finansijske i životne gubitke kod potrošača). Takođe, može se predvideti gde će doći do sledećeg prekida.

Preko pametnih brojila i prognoziranja potrošnje može se predvideti preopterećenje dela mreže, brzo delovati i sprečiti potencijalni ispadi u napajanju.

Mogu se koristiti podaci i analiza koja može predvideti rast vegetacije na osnovu tipa zemljišta, tipa vegetacije i nivoa padavina. Na taj način se mogu identifikovati potencijalna rizična područja za kontakte drveća sa prenosnim ili distributivnim vodovima i na taj način sprečiti poremećaj u mreži pre nego što se desi.

Dodatna upotreba podataka o preletu iz dronova za mapiranje palih stabala, oborenih vodova i poplavnih područja mogu se koristiti za detekciju, predviđanje kvarova, kao i za optimizaciju restauracije napajanja.

Kao društveni senzor u realnom vremenu za pametnu mrežu, društveni mediji kao što su Twitter ili Fejsbuk mogu sadržati potencijalne informacije koje ukazuju na pojavu nestanka struje u određenom području i njegovo efikasno lociranje.

Predviđanje kvara opreme

Na osnovu SCADA podataka, relejne zaštite, senzora, i podataka o broju okidanja prekidača i vremenu okidanja (da li traje predugo) može se identifikovati prekidač koji

će verovatno ubrzo propasti i unapred isplanirati njegova zamena i tako sprečiti budući kvar. Ili da se u toku rada, iskoristi za manipulaciju drugi prekidač koji ima veći preostali kapacitet prekidanja.

Podaci dobijeni iz preleta dronova mogu pomoći da se identifikuju oštećeni vodovi i oštećeni izolatori bez ugrožavanja zaposlenih. GIS i slikovni podaci mogu se koristiti za identifikaciju problema. Npr. život transformatora se može predvideti posmatranjem modela opterećenja tokom godine.

Električna vozila

Električna vozila (EV) pune svoje baterije preko mreže, što nameće značajan uticaj na mreže elektroenergetskih sistema. Na primer, punjenje EV-a u naseljenom području tokom vremena vršne potrošnje može imati posledice kao što su ispadanje osigurača, smanjena efikasnost i degradacija transformatora. Kroz svoju dvosmernu komunikacionu tehnologiju, pametne mreže mogu ovo rešiti tako što će raspoređivati punjenje EV-a za časove van vršne potrošnje. A to će omogućiti veliki podaci kroz formiranja obrazaca punjenja i praznjenja EV-a.

Primene u relejnoj zaštiti

Relej je uređaj koji služi za detekciju nenormalnih pogonskih stanja uređaja i delova elektroenergetskih sistema i inicijalizaciju odgovarajućih upravljačkih akcija za obezbeđenje normalnog pogona [8]. Analiza velikih podataka, koji pristižu iz različitih izvora, mogu uticati na podešenje relejne zaštite. Taj uticaj ima za razultat poboljšanje reagovanja releja na poremećaje u mreži, ostvarujući na taj način veću pouzdanost i sigurnost u sistemu.

Jedan od primera gde bi primena velikih podataka i adaptivne zaštite dala veoma dobre rezultate je termička zaštita elemenata elektroenergetskog sistema. Ako bi se pored vrednosti struje, analizirali i podaci o temperaturi okoline i brzini vetra, moglo bi se u određenim uslovima (hladno vreme sa vetrom), bez posledica po štice element, dopustiti i trajno preopterećenje elementa.

Analizom podataka, koji su potrebni za predviđanje potrošnje potrošača u određenom periodu i podataka o nastupajućem društvenom dešavanju koje se dešava i traje određeno kratko vreme, može se dopustiti preopterećenje elemenata (fider, transformator) u mreži. Na osnovu toga može se prepodesiti relejna prekomerna zaštita na taj određeni period kako ne bi delovala zbog manjeg prekoračenja podešene vrednosti. Sve pod uslovom da takvo preopterećenje neće drastično uticati na stanje elemenata. Na osnovu podataka o istorijskom i trenutnom stanju elemenata, predviđenom životnom veku, isplativosti remonta, odredilo bi se da li je takva odluka efikasna, dovoljno bezbedna za stanje elemenata i isplativa za samo preduzeće.

Izazovi za analitiku velikih podataka su višestruki. Postoje i tehnički izazovi koji proističu iz karakteristika elektroenergetskog sistema i netehnički izazovi. Pre svega, treba prvo napomenuti da većina elektroenergetskih kompanija još uvek nije sigurna u analitiku velikih podataka, mašinsko učenje i veštačku

inteligenciju [4]. Potrebne su korenite promene. Izazovi su:

- dominacija energetske elektronike
- standardizacija za primenu tehnika velikih podataka
- privatnost i zaštita poverljivosti podataka
- obuka stručnjaka za podatke
- regulative i zakoni

5. ZAKLJUČAK

Tehnologija velikih podataka u elektroindustriji dobija bitnu ulogu koja će omogućiti ubzan razvoj, bolje planiranje i efikasniji rad elektroenergetskog sistema, kao i trenutno prilagođavanje novonastalim događajima i sve zahtevnijim potrebama potrošača. Funkcionisanje moderne električne mreže i upravljanje fleksibilnijom proizvodnjom/potrošnjom, neće biti moguće bez odgovarajućih naprednih komunikacija i tehnologija obrade i analize velikih podataka, koje su u stanju da prikupe ogromnu količinu heterogenih podataka i obrade u realnom vremenu. Primenom analitike i veštačke inteligencije, sistem će automatizovano donositi najbolje odluke i proaktivno delovati, kako bi održavao stabilan rad, smanjio posledice i predvidio stanje kvara, sačuvao opremu mreže i obezbedio efikasnije korišćenje električne energije. Sve ovo donosi korist, kako elektroprivredi i samom elektroenergetskom sistemu u vidu smanjenja troškova i pouzdanijeg rada, tako i samim potrošačima u pogledu njihovog zadovoljstva kvalitetom napajanja.

6. LITERATURA

1. M.Henderson: *We Have the Data: Making It Work for Us*, IEEE Power and Energy Magazine, Vol.16, No.3, pp.4–100, May 2018.
2. N.Amaro, J.M.Pina: *Big Data in Power Systems*, IEEE, International Conference on Engineering, Technology and Innovation (ICE/ITMC), Funchal, Portugal, 2017.
3. Working Group on Big Data Analytics, Machine Learning and Artificial Intelligence in the Smart Grid: *Big Data Analytics in the Smart Grid*, White Paper, IEEE Smart Grid
4. L.Xie: *Data Technology: The New Normal*, IEEE Power and Energy Magazine, Vol.16, No.3, pp.108–106, May 2018.
5. J.Puurtinen: *Big data mining as a part of substation automation and network management*, Master of Science Thesis, Tampere University of Technology, May 2014.
6. D.Cai, H.Tian, Y.Wang, H.Wang, H.Zheng, K.Cao, C.Zhou: *Electric Power Big Data and Its Applications*, Atlantis Press, 2016.
7. Y.Zhang, T.Huang, E.F.Bompard: *Big data analytics in smart grids: a review*, Energy Informatics, Vol.1, No.8, 2018.
8. D.Bekut: *Relejna zaštita*, Fakultet Tehničkih Nauka u Novom Sadu, Novi Sad, 2009.

Kratka biografija:



Stefan Dorđević rođen je u Smederevu 1992. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Elektroenergetski sistemi odbranio je 2019. god.

**RAZVOJ I UPOREDNA ANALIZA PROGRAMSKIH JEZIKA ZA PAMETNE UGOVORE
NA BLOKČEJNU****THE DEVELOPMENT AND COMPARATIVE ANALYSIS OF PROGRAMMING
LANGUAGES FOR BLOCKCHAIN-BASED SMART CONTRACTS**

Nebojša Horvat, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu opisani su osnovni koncepti distribuiranih sistema, blokčejn tehnologije i programskih jezika za pisanje pametnih ugovora. Razvoj blokčejn tehnologije je podeljen na tri generacije od kojih je svaka donela nove jezike za pisanje pametnih ugovora. Jezici su zasebno analizirani i na kraju je izvršena njihova komparativna analiza u cilju određivanja poželjnih osobina programskih jezika za pisanje pametnih ugovora.

Abstract – In this work we present the fundamentals of distributed systems, blockchain technology and languages for writing smart contracts. The development of blockchain technology was divided into three generations out of which the each one has brought some new languages for writing smart contracts. Programming languages were then separately analyzed and then a comparative analysis was done in order to determine the desired properties of languages for writing blockchain smart contracts.

Ključne reči: distribuirani sistemi, blokčejn, pametni ugovori.

1. UVOD

Krajem dvadesetog veka, relacije baze podataka postaju nezaobilazni činilac informacionih sistema. One se oslanjaju na sisteme za upravljanje bazama podataka i njihova evidencija transakcija omogućava analizu podataka. Baze podataka su na početku bile pretežno centralizovane. Takve baze lakše je konstruisati i održavati, ali imaju velika ograničenja sa aspekta bezbednosti zato što baza predstavlja jedinstvenu tačku otkaza. Takođe, centralizovane baze je veoma teško skalirati. Iz navedenih razloga, i usled porasta količine podataka koje je potrebno skladištiti, dolazi do razvoja distribuiranih baza podataka. Kod distribuiranih baza može se iskoristiti memorijski kapacitet i procesorska moć svih računara na kojima je baza distribuirana. Oporavak od otkaza dela baze mnogo je lakši nego kod centralizovanih baza, gde otkaz baze često znači i gubitak podataka.

Distribuirane baze podataka rešavaju pomenute probleme u čuvanju podataka. Međutim, u poslovanju između dve stranke postoji dodatni skup problema koje je potrebno rešiti.

U većini slučajeva, transakcije prolaze niz koraka kako bi se utvrdila njihova validnost. Najveći problem je održavati

organizaciju od poverenja koja bi bila zadužena za validaciju transakcija. Takva organizacija bila bi posrednik u poslovanju stranaka. Odličan primer takve organizacije bile bi banke koje su posrednici u finansijskim transakcijama. Loša strana pomenutih rešenja je to što postoji posrednik u koga obe strane moraju imati poverenja i on je odgovoran za pravilan rad čitavog sistema.

Rešenje svih prethodno navedenih problema se nalazi u upotrebi blokčejn (engl. *blockchain* - lanac blokova) tehnologije. Blokčejn sistemi omogućavaju da se poslovanje odvija bez posrednika uz očuvanje prednosti koje pružaju distribuirani sistemi. U zavisnosti od domena problema, moguće je koristiti privatne i javne blokčejn sisteme. Javni sistemi uvode transparentnost i mogućnost svakog člana mreže da validira transakciju i lokalno čuva spisak svih transakcija. Ovakvi sistemi pružaju sigurnost, otpornost na otkaze i povećanje brzine izvršavanja transakcija. Dok je javni blokčejn otvoren i omogućava svakome da učestvuje u radu sistema, privatni blokčejn omogućava samo ovlašćenim članovima da učestvuju u radu zatvorene mreže. Svaki član ima prava i ograničenja u mreži i ona se razlikuju od člana do člana. Privatni blokčejn sistemi ograničavaju pristup podacima, te su zato idealni za konzorcijum organizacija koje međusobno sarađuju.

Prva generacija blokčejn tehnologije, poput Bitcoin [1], bila je ograničena na to da podrži samo kriptovalute uz vrlo malo mogućnosti za pravljenje drugih sistema zasnovanih na njima. Naredna generacija tehnologije, pokrenuta Ethereumom [2] i Hyperledger Fabricom [3] je uvela programsku podršku za koncept pametnih ugovora. Uz pomoć pametnih ugovora možemo da pravimo proizvoljne aplikacije zasnovane na blokčejn sistemima.

Cilj ovog rada je da se predstavi trenutno stanje u razvoju blokčejn tehnologije sa posebnim fokusom na pametne ugovore i jezike u kojima se oni pišu. Za ispunjenje pomenutog cilja potrebno je realizovati nekoliko koraka. Najpre je potrebno izvršiti analizu teorijskih osnova blokčejn tehnologije. Pod tim se podrazumeva analiza koncepta i tehnologija koje su neophodne kako bi blokčejn sistemi funkcionisali. Takođe, važno je naglasiti i objasniti kako pomenute tehnologije u međusobnoj interakciji pružaju prednosti blokčejn sistema kao što su distribuiranost, sigurnost, transparentnost, itd. Nakon toga je potrebno analizirati jezike koje svaki sistem pruža za pisanje pametnih ugovora i izvršiti uporednu analizu pronađenih jezika.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dušan Gajić, docent.

2. DISTRIBUIRANI SISTEMI

Distribuirani sistem je skup autonomnih računara koji čine celinu i iz perspektive korisnika izgledaju kao jedan koherentan sistem [4]. Prethodni iskaz obuhvata dve glavne osobine distribuiranih sistema. Prva ističe da je distribuirani sistem skup računarskih elemenata koji su nezavisni jedni od drugih; takvi elementi se najčešće nazivaju čvorovi (engl. node - čvor). Druga osobina obuhvata korisnike koji mogu biti ljudi ili aplikacije. Korisnici pri interakciji sa sistemom moraju imati osećaj kao da komuniciraju sa jedinstvenom aplikacijom, a ne sa skupom čvorova. Ovo znači da čvorovi moraju intenzivno da komuniciraju i sarađuju. Način ostvarivanja takve komunikacije među čvorovima je srž distribuiranog sistema.

Kako korisnik može da komunicira isključivo sa celim sistemom, on ne zna od kojih elemenata tj. čvorova se sistem sastoji. Podaci su takođe raspoređeni po različitim čvorovima. To znači da korisniku koji se obraća sistemu mesto čuvanja podataka nije važno, te je sistem dužan samo da obezbedi podatke. Ovo dovodi do česte replikacije podataka kako bi se pružile bolje performanse i to nazivamo transparentnošću distribuiranosti (engl. distribution transparency).

3. TEHNOLOGIJA DISTRIBUIRANE GLAVNE KNJIGE I BLOKČEJN

Distribuirana glavna knjiga ili tehnologija distribuirane glavne knjige (engl. distributed ledger technology - DLT) je vrsta distribuirane baze podataka koja pretpostavlja moguće prisustvo malicioznih korisnika (čvorova). DLT je, takođe, vrsta strukture podataka za pamćenje transakcija koje su razmeštene na više čvorova (fizičkih mašina). Ključno je da su čvorovi povezani u peer-to-peer mrežu gde svaki čvor sadrži identičnu repliku glavne knjige. Takođe, svaki čvor samostalno osvežava svoje podatke, tj. kopiju glavne knjige. Za razliku od centralizovane glavne knjige koja obično ima regulatorno telo, kod distribuirane glavne knjige svaki čvor može da upiše novu transakciju i zatim ostali čvorovi glasaju kroz konsenzus algoritam o tome koja kopija glavne knjige je validna. U trenutku kada je konsenzus postignut, svi čvorovi upisuju validnu verziju glavne knjige.

Blokčejn je distribuirana struktura podataka koja implementira distribuiranu glavnu knjigu, koja je sastavljena od lanca kriptografski povezanih blokova koji sadrže grupe transakcija. U opštem slučaju, vrši se emitovanje (engl. broadcast) svih podataka svim učesnicima u mreži. Glavna razlika između blokčejna i drugih distribuiranih baza podataka je u tome što je blokčejn projektovan kako bi se postigao konzistentan i pouzdan dogovor o zapisu događaja (npr. „ko je vlasnik čega“) između nezavisnih učesnika koji mogu imati različite motivacije i ciljeve.

3.1. Generacije blokčejn tehnologije

Ukoliko posmatramo razvoj blokčejn sistema od njihovih zvaničnih početaka 2008. godine kada je Satoshi Nakamoto objavio rad o Bitcoinu, mogli bismo uvideti tri faze u razvoju ovih sistema. Prva faza bi svakako bila dominacija bitcoina i drugih kriptovaluta. Kriptovalute su bile do te mere dominantne da ljudi nisu ni znali da

blokčejn može da se koristi u druge svrhe. Druga generacija ili faza razvoja bila bi pojava Ethereuma, koji koristi blokčejn tehnologiju, ne samo u svrhu pravljenja kriptovalute, već kao svetski kompjuter (engl. world computer) na kome mogu na bezbedan način da se izvršavaju proizvoljni programi (pametni ugovori). Treća generacija bili bi privatni blokčejn sistemi sa kontrolom pristupa, koji se koriste u poslovanju različitih organizacija. Primer bi bili Hyperledger Fabric i Tendermint koji pružaju platformu za razvoj poslovnih rešenja baziranih na blokčejn tehnologiji.

Kao što je već naglašeno, u **prvoj generaciji** blokčejna kriptovalute su bile dominantne. Bitcoin se posebno ističe među njima kao prva i najpopularnija kriptovaluta. Kriptovaluta je sistem za digitalni transfer vrednosti. Bitcoin se koristi za trgovinu i akumulaciju vrednosti. Korisnici mreže međusobno komuniciraju kroz Bitcoin protokol. Komunikacija se pretežno odvija preko interneta. Bitcoin protokol i sve usko povezane aplikacije su dostupne kao softver otvorenog koda i mogu se pokrenuti na velikom broju različitih računara uključujući i pametne telefone koji su danas široko pristupačni.

Ono što je Ethereum, kao predstavnik **druge generacije**, doneo kao radikalnu promenu jeste pružanje blokčejn platforme sa ugrađenim Turing kompletnim programskim jezikom koji može biti iskorišćen za pisanje pametnih ugovora. Takvi ugovori mogu da podrže proizvoljne transakcije za prenos stanja ili omogućavaju korisnicima da naprave proizvoljan sistem koji koristi blokčejn platformu. Proizvoljni sistemi se mogu realizovati kroz prosto pisanje pametnih ugovora u nekoliko linija koda.

Pomenute prednosti Ethereuma omogućavaju brz razvoj aplikacija, sigurnost za male aplikacije koje nemaju dovoljno resursa da brinu o bezbednosti i mogućnost da male aplikacije vrlo efikasno komuniciraju. Takve funkcionalnosti se postižu kroz kreiranje apstraktnog korenskog sloja na kome se sve aplikacije izvršavaju kroz pametne ugovore napisane u Turing kompletnom jeziku. Ovo omogućava svima da na lak i brz način napišu pametne ugovore koji predstavljaju decentralizovanu aplikaciju koja ima svoja proizvoljna pravila vlasništva.

U **treću generaciju** ubrajamo privatne blokčejn sisteme sa kontrolom pristupa poput Hyperledger Fabrica, Hyperledger Sawtooth i Tendermint koji pružaju platformu za razvoj poslovnih aplikacija. Hyperledger Fabric je blokčejn sistem sa kontrolom pristupa napravljen da zadovolji potrebe poslovnih aplikacija [5]. Otvorenog je koda i ima izuzetno modularnu i konfigurabilnu arhitekturu koja može da se prilagodi potrebama velikog broja poslovnih slučajeva korišćenja. Jedna od najvećih prednosti Fabrica je to što omogućava pisanje pametnih ugovora u programskim jezicima opšte namene kao što su Go, Java i JavaScript. Značajan aspekt Fabric mreže je da je identitet korisnika u mreži poznat i da dva učesnika ne moraju da veruju jedan drugom da bi uspešno poslovali. Dovoljno je da se njihovo poslovanje formalno definiše i mreža će voditi računa o tome da su svi aspekti dogovora ispoštovani.

4. RAZVOJ PROGRAMSKIH JEZIKA ZA PISANJE PAMETNIH UGOVORA

Pored arhitekture samog sistema, koja nije previše podložna promeni, pametni ugovori predstavljaju najvažniji element svakog blokčejn sistema za programere koji ga razvijaju. Kroz pametne ugovore se realizuje sva poslovna logika sistema i zato je važno da oni budu napisani na što kvalitetniji način, kako bi što bolje oponašali poslovne procese. Međutim, kako je sva logika sistema koncentrisana u pametnim ugovorima, njihovo pisanje predstavlja proces koji zahteva najviše resursa. Pametni ugovori, na primer, mogu biti zaduženi za prebacivanje velike količine novca, tako da bilo kakva greška u njihovom kodu može dovesti do velikih i neprihvatljivih gubitaka.

Kada govorimo o **I generaciji** blokčejn sistema, najčešće podrazumevamo Bitcoin tehnologiju, kao prvu i dominantnu kriptovalutu. Bitcoin koristi Script jezik za pisanje pametnih ugovora. Script je sličan Forth programskom jeziku, koristi obrnutu poljsku notaciju i baziran je na steku. Kada kažemo da je jezik baziran na steku, to u ovom slučaju znači da ne postoje registri u kojima se čuvaju operandi ili upisuju rezultati operacija, već se sve smešta na stek. Kod obrnute poljske notacije, operandi prethode operatoru, što uklanja potrebu za zagradama.

Script je veoma jednostavan programski jezik koji je dizajniran tako da bude ograničen i lak za pokretanje na različitim hardverskim platformama čak i na ugrađenim računarskim sistemima. Potrebna je minimalna količina procesne moći da bi Script radio, ali mnoge stvari koje programski jezici opšte namene podržavaju, u njemu ne mogu biti napisane. Sva ograničenja koja nosi sa sobom su namerno postavljena od strane njegovih kreatora i služe radi povećanja bezbednosti sistema.

Glavna novina koju **II generacija** donosi sa sobom je mogućnost pisanja pametnih ugovora u Tjuring kompletnom jeziku koji nema toliko izražena ograničenja kao što je slučaj kod Bitcoinovog Script jezika. Ono što je omogućilo Ethereumu da koristi Tjuring kompletan jezik jeste uvođenje virtualizacije, tj. Ethereum virtuelne mašine (engl. Ethereum Virtual Machine - EMV) i uvođenje pojma gasa. Svaka instrukcija izvršena u pametnom ugovoru troši gas i tako sprečava napade koji onemogućavaju rad sistema (engl. Denial of Service - DoS). Ethereum programeri mogu da koriste programske jezike višeg nivoa kao što su Solidity, Serpent ili LLL. Serpent je jezik niskog nivoa koji je baziran na Pythonu, LLL je takođe jezik niskog nivoa baziran na Lispu i oba se retko koriste. Solidity je objektno orijentisan jezik visokog nivoa koji je specifično dizajniran za pisanje pametnih ugovora. Dizajniran je po ugledu na JavaScript, C++, Python i maksimalno je optimizovan da dobro radi na Ethereum virtuelnoj mašini. Solidity ima strogo definisane tipove, podržava nasleđivanje i omogućava upotrebu biblioteka i pravljenje proizvoljnih korisničkih tipova podataka.

U **III generaciji** uvodi se mogućnost pisanja pametnih ugovora kroz upotrebu jezika opšte namene. Ovo je značajna prednost, pre svega jer programeri koji razvijaju pametne ugovore ne moraju više da uče novi DSL (engl. Domain Specific Language - jezici specifični za domen)

kao što je slučaj kod Bitcoina ili Ethereumu. Fabric trenutno omogućava pisanje pametnih ugovora u Golangu, JavaScriptu i Javi. Sva tri navedena jezika su izuzetno popularna tako da postoji velika verovatnoća da će programer biti ekspert u jednom od navedenih jezika i tako vrlo brzo ovladati tehnikom za pisanje kvalitetnih pametnih ugovora.

5. UPOREDNA ANALIZA PROGRAMSKIH JEZIKA ZA PISANJE PAMETNIH UGOVORA

Blokčejn sistemi su sve zastupljeniji te se javlja potreba za pisanjem velikog broja pametnih ugovora. Problem kod blokčejn programiranja je to što se ugovori moraju veoma pažljivo pisati. I najmanja greška može da dovede do pada celog sistema. Danas možemo reći da smo došli do treće generacije ovih kompleksnih sistema, tako da ćemo se osvrnuti na to kako je izgledao razvoj njihovih jezika i šta je svaka nova generacija donela kao poboljšanje.

5.1. Tjuring kompletnost

Od svih jezika koje smo obradili do sada, jedino Script nije bio Tjuring kompletan; on podržava grananje ali ne podržava „goto” naredbu tako da se u njemu ne mogu napraviti petlje. Ovo je ozbiljno ograničenje svakog programskog jezika. Script je s namerom dizajniran sa značajnim ograničenjima kako bi se povećala sigurnost pametnih ugovora. Bitcoin nema napredne sigurnosne mehanizme kao sledeće generacije, tako da je njegov sigurnosni mehanizam to što je jezik za pisanje pametnih ugovora ekstremno ograničen, te je jako teško napraviti maliciozan kod. Svi drugi programski jezici, počevši od Solidity-a i Serpenta, pa do Golang, Jave i JavaScripta su Tjuring kompletni.

5.2. Ekspresivnost

Ekspresivna moć programskog jezika predstavlja raznovrsnost i širinu ideja koje se mogu predstaviti i napisati u tom jeziku. Krenućemo od Scripta, kako je on prvi programski jezik za pisanje pametnih ugovora i svakako ima najmanju ekspresivnu moć. Mnoge ideje se u njemu ne mogu predstaviti. Svaki sistem koji u sebi sadrži koncept ponavljanja proizvoljni broj puta, nije moguće konstruisati u Scriptu.

Solidity je objektno orijentisan, Tjuring kompletan jezik koji sa sobom donosi mnogo više funkcionalnosti u poređenju sa Scriptom. U njemu je moguće opisati eksponencijalno više sistema nego u Scriptu; međutim, i on sa sobom nosi određene nedostatke. Za razliku od drugih modernih programskih jezika, Solidity je napravljen tako da je iz njega izbačen nedeterminizam. Determinizam povećava sigurnost koda i smanjuje mogućnost za grešku, ali smanjuje ekspresivnost jezika.

Iako je Go najnoviji programski jezik koji se može koristiti za pisanje pametnih ugovora na Fabricu, čini se da je on najmanje ekspresivan. Činjenica da su iz Go-a izbačeni koncepti poput nasleđivanja i generalizacije, u značajnoj meri ograničava njegovu ekspresivnost.

Java i JavaScript su najstariji i najzreliji jezici od svih spomenutih. Njihove mogućnosti su izuzetne, objektno su orijentisani, podržavaju funkcionalno programiranje, konkurentnost i sve druge koncepte koji se očekuju od programskog jezika opšte namene. Tako da su oni najekspresivniji jezici za pisanje pametnih ugovora.

5.3. Performanse

Kada se porede performanse različitih programskih jezika za pisanje pametnih ugovora koji se koriste na različitim blokčejn sistemima, vrlo je teško izolovati performanse samog sistema od performansi programskog jezika. Takođe je vrlo važno naglasiti da se kod programskih jezika na blokčejn sistemima mnogo više ceni da količina podataka koja treba da se upiše u glavnu knjigu bude mala, nego da se podaci brzo upišu. Oprez pri skladištenju podataka je pogotovo izražen kod javnih blokčejn sistema gde se svi podaci čuvaju u jednoj velikoj glavnoj knjizi čija veličina može značajno da poraste vremenom i tako opteretiti svaki čvor u mreži.

Script je izuzetno jednostavan jezik i vrlo je niskog nivoa. On pruža dobre performanse pri pokretanju pametnih ugovora. S druge strane, važno ograničenje Scripta je to što on čini komplikovanije pametne ugovore napisane u njemu izuzetno glomaznim. Ovo je veliki problem s obzirom da Bitcoin ima ograničenje veličine njegovih skripti.

Solidity je jezik visokog nivoa koji je dodatno interpretiran na Ethereum virtuelnoj mašini. Kod jezika koji se kompajliraju na EVM, najbitniji podatak je koliko gasa troše pri izvršavanju pametnih ugovora. Ethereum mnogo naplaćuje ugovorima koji su spori i zahtevaju mnogo računanja, tako da je značajno da programski jezik pruža maksimalne performanse. Kako je Solidity programski jezik najvišeg nivoa dostupan za pisanje pametnih ugovora na Ethereumu, to ga čini najlakšim za pisanje i najpopularnijim, ali ujedno i programskim jezikom sa najlošijim performansama na Ethereum mreži.

Što se tiče programskih jezika za pisanje pametnih ugovora na Hyperledger Fabricu, kod njih prostor nije veliki problem, s obzirom na to da je privatni blokčejn uglavnom mnogo manje opterećen. Golang bi bio prirodan izbor u potrazi za jezikom koji pruža najbolje performanse. On se kompajlira i njegov kod se direktno izvršava, za razliku od Jave za čije izvršavanje je potrebna JVM ili JavaScripta, koji je interpretirani jezik i stoga mnogo sporiji.

5.4. Bezbednost

Bezbednost je apsolutno najbitniji aspekt pri pisanju pametnih ugovora, tako da bi trebalo da je prioritet pri izboru programskog jezika.

Kao što je već rečeno u ranijem tekstu, jedina sigurnost koju Bitcoin pruža pri izvršavanju svojih skripti jeste ograničenost njegovog jezika. To čini Script ubedljivo najbezbednijim jezikom za korišćenje.

Solidity je programski jezik koji je napravljen sa mnogim ograničenjima. Razlozi za namerno kreiranje svih opisanih ograničenja leže baš u potrebi za povećanjem bezbednosti. Ukoliko uklonimo nedeterminizaciju, povećavamo bezbednost sistema i prostor za pojavu slučajnih grešaka.

Od spomenutih jezika opšte namene, Go je svakako najbezbedniji. Ograničenja koja su uvedena u Go su pretežno tu da bi povećala čitljivost i čistoću koda. Bezbednost Go-a i dalje nije na nivou koji pruža Solidity ili pak Script, ali je on znatno bezbedniji od Jave, a pogotovo od JavaScripta koji nema ni strogo definisane tipove.

6. ZAKLJUČAK

U prethodnim segmentima rada je opisana tehnologija koja se nalazi iza blokčejn sistema, kao i njihov razvoj koji je

podeljen u tri generacije. Važno je primetiti da su kroz generacije sistemi dizajnirani s različitim idejama o njihovoj upotrebi. Bitcoin je bio zamišljen kao kriptovaluta i siguran način za razmenu novca bez potrebe za regulatornim telom. Ethereum je prepoznao mogućnosti i prednosti koje blokčejn tehnologija donosi sa sobom i težio da napravi svetski računar (engl. world computer). Ethereum je zamišljen kao platforma koja bi omogućila lak i brz razvoj blokčejn aplikacija koje bi se realizovale preko pametnih ugovora. Za razliku od Bitocina, jezik za pisanje pametnih ugovora kod Ethereuma je Tjuring kompletan i pruža mnogo veće mogućnosti od Scripta. Sledeća faza razvoja su privatni blokčejn sistemi sa kontrolom pristupa. Povećana popularnost blokčejn sistema je dovela do želje da se oni upotrebe u poslovnim aplikacijama koje imaju drugačije potrebe, koje se ne mogu zadovoljiti javnim blokčejn sistemima. Poslovne strane često zahtevaju postojanje identiteta i zabranu pristupa, tako da samo klijenti sa dozvolom mogu da učestvuju u poslovnom procesu.

Kroz razvoj blokčejn sistema, jasno se vidi da je stvar koja se najviše menjala zapravo jezik u kome su pametni ugovori pisani. Kako su pametni ugovori jedan on najbitnijih aspekata sistema i svakako deo sistema koji sadrži poslovnu logiku, važno je rukovati programskim jezikom u kome se ta logika može jednostavno i kvalitetno implementirati. Script je veoma jednostavan programski jezik u kome je bezbedno pisati pametne ugovore i lakše ih je formalno verifikovati ali je veoma neekspresivan, što ga čini veoma teškim za rad. Solidity je Tjuring kompletan i mnogo ekspresivniji od Scripta, ali i on sa sobom nosi mnoga ograničenja koja su napravljena radi njegove sigurnosti. Velika mana i Solidity-a i Scripta je to što su oni DSL-ovi. Golang, Java i JavaScript su prvi jezici opšte namene koji su korišćeni za pisanje pametnih ugovora. Rad sa popularnim jezicima je vrlo olakšavajući zato što su oni dobro poznati programerima i tako se brže može doći do kvalitetnih pametnih ugovora.

7. LITERATURA

- [1] Nakamoto S., *Bitcoin: A Peer-to-Peer Electronic Cash System*, dostupno na: <https://bitcoin.org/bitcoin.pdf> poslednji pristup 20.08.2019.
- [2] Vitalik Buterin, 2013. *A Next-Generation Smart Contract and Decentralized Application Platform*, dostupno na: <https://github.com/ethereum/wiki/wiki/White-Paper>, poslednji pristup 20.08.2019.
- [3] Androulaki, E., Barger, A., Bortnikov, V., Cachin, 2018, April. Hyperledger fabric: a distributed operating system for permissioned blockchains
- [4] Tanenbaum, A.S. and Van Steen, M., 2017. *Distributed systems: principles and paradigms*. Prentice-Hall., <https://www.distributed-systems.net/index.php/books/distributed-systems-3rd-edition-2017/>, poslednji pristup 20.08.2019.
- [5] Hyperledger Fabric dokumentacija, <https://hyperledger-fabric.readthedocs.io/en/release-1.4>, poslednji pristup 20.08.2019.

Kratka biografija:



Nebojša Horvat je rođen u Novom Sadu, 1995. god. Fakultet tehničkih nauka upisao je 2014. god. Osnovne akademske studije završio je 2018. god.

**АДАПТИВНО УПРАВЉАЊЕ КРАНОМ ПРИМЈЕНОМ „GAIN-SCHEDULING”
АЛГОРИТМА****A GAIN-SCHEDULING APPROACH FOR ADAPTIVE CRANE SYSTEM CONTROL**

Ања Буљевић, Факултет техничких наука, Нови Сад

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – Тема овог рада јесте пројектовање адаптивног регулатора за управљање 3Д краном. Задатак овог регулатора јесте да управља позицијом крана уз стабилизацију терета током кретања тј. смањења његових осцилација. С обзиром на промјенљиву дужину клатна на ком је објешен терет, мијења се и динамика система, па је због тога неопходно имплементирати адаптивни регулатор. Адаптивни алгоритам који је кориштен у овом раду је gain-scheduling (ГС). Имплементирани су следећи регулатори: линеарни регулатор по стањима, линеарни регулатор по излазима и линеарни регулатор по излазима са интегралним дејством, а затим су сви њихови одзиви упоређени.

Кључне речи: 3Д кран, адаптивни регулатор, ГС

Abstract –The main purpose of this paper is designing adaptive controller for controlling 3D Crane system. This controller has a task to control a position of this crane providing that the payload on the crane makes as smaller as possible oscillations. The payload is hanging on pendulum which length can be changed and due to this fact it is necessary to implement adaptive controller. Adaptive algorithm that was used in this paper is called gain-scheduling (GS). Following controllers are implemented: linear state-space controller, linear output feedback controller and linear output feedback controller with integral component, and after that their responses are compared.

Keywords: 3D crane, adaptive controller, GS

1. УВОД

Кранови се користе у многим индустријама, фабрикама и магацинима како би пренијели терете са једног мјеста на друго. Иако у пракси најчешће срећемо човјека како управља овим машинама, аутоматизовани системи би били доста безбједнији за човјека, али и прецизнији.

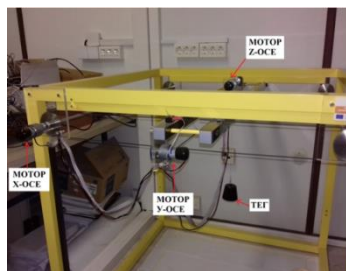
У литератури се могу пронаћи различити математички модели 3Д крана, као и различите технике за управљање краном [1], [2].

Inteco® 3D Crane [3] је лабораторијски модел 3Д крана чији се рад контролише уз помоћ рачунара.

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био др Мирна Капетина, доцент.

Он представља нелинеарни електромеханички систем који има комплексно динамичко понашање и ствара занимљиве и изазовне проблеме приликом управљања. Изглед 3Д крана је приказан на слици 1.



Слика 1. Изглед 3Д крана

Први задатак овог рада био је формирање математичког модела 3Д крана који би на адекватан начин описивао динамику реалног система. Након тога је било неопходно естимирати непознате параметре модела и естимација је урађена методом најмањих квадрата [4]. Централно мјесто у овом раду заузима пројектовање адаптивног управљачког алгоритма који би обезбједио помјерање терета до жељене позиције, његову стабилизацију током промјене положаја и смањење његових осцилација. Проблем се јавља када се мијења дужина клатна јер се тада параметри модела мијењају, па самим тим и параметри регулатора који су подешени за једну висину нису више одговарајући. Проблем је ријешен примјеном адаптивног управљања који се заснива на регулатору са промјенљивим (табличним) појачањем (gain-scheduling, ГС)

**2. ФОРМИРАЊЕ МАТЕМАТИЧКОГ МОДЕЛА
3Д КРАНА**

Као што је већ напоменуто, кран је веома нелинеаран комплексан динамички систем уколико се посматра са свим својим величинама и промјенљивом дужином клатна што изазива доста проблема у управљању. Међутим, за потребе управљања није неопходно посматрати такав сложен систем у цјелини, већ је довољно да се полазним нелинеарним систем опише поједностављеним линеарним моделом. За потребе овог рада описана је динамика кретања крана по једној оси и формиран је следећи математички модел

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= a_{22}x_2 + b_2u \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= a_{43}x_3 + b_4u \end{aligned} \quad (1)$$

гдје је:

x_1 - удаљеност шина са колицима од нулте позиције рама ,

x_2 - брзина кретања шина са колицима,

x_3 - угаони помјерај,

x_4 - угаона брзина.

Прве двије једначине модела описују динамику кретања крана по позицији, а друге двије једначине дају динамику кретања угла крана. Примјећујемо да су једначине које описују кретање по позицији и отклон угла међусобно независне.

3. ЕСТИМАЦИЈА ПАРАМЕТАРА 3Д КРАНА

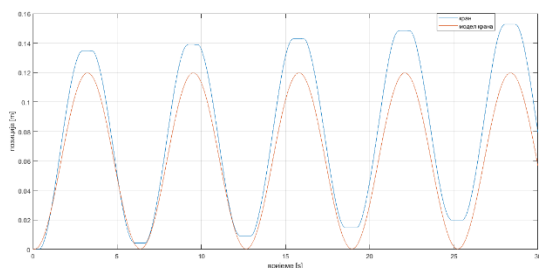
Естимација параметара подразумева проналазак непознатих параметара датог система. У овом раду је урађена офлајн естимација параметара и параметри који су требали да се идентификују су a_{22} , a_{43} , b_2 , b_4 за x и y -осу. Као што је већ напоменуто, у моделу крана (1) позиција не утиче на угао, као ни угао на позицију. Због ове чињенице, модел је посматран из два дијела па су параметри a_{22} и b_2 естимирани независно од параметара a_{43} и b_4 . 3Д кран има промјенљиву дужину клатна и у зависности од дужине клатна, параметри математичког модела се мијењају, па је естимација параметара одрађена за 3 различите дужине: $l=0.05m$, $l=0.278m$ и $l=0.5m$. Метод који је кориштен за естимацију параметара a_{22} , b_2 , који се односе на дио модела кретања крана по позицији, је метод најмањих квадрата [5]. Алгоритам је такав да захтјева богат побудни сигнал, тако да је на улаз система доведен сигнал сљедећег облика $u(t) = 0.2 \sin(\pi t + 0.23) + 0.3 \sin(0.2\pi t)$. (2) Након нумеричке симулације добијени су сљедећи резултати за x -осу и y -осу (Табела 1.).

Табела 1. Добијени естимирани параметри за x и y осу за $l=0.278m$

	a_{22}	b_2
x	-10	2
y	-6	2.75

Дужина клатна не утиче на промјену непознатих параметара модела крана који описује кретање крана по позицији, тако да за било коју дужину клатна, математички модел позиције остаје исти.

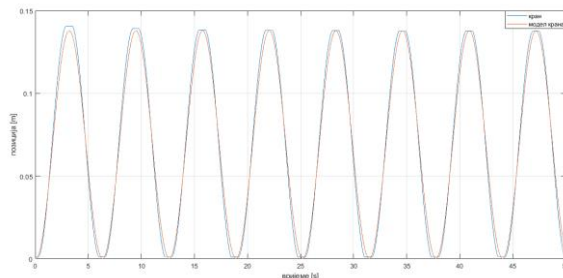
На сликама 3. и 4. је дат упоредни приказ понашања 3Д крана и његовог математичког модела за x и y -осу када је дужина клатна $l=0.278m$. Понашање система је слично и за остале дужине клатна, па ти одзиви неће бити приказани у наставку.



Слика 3. Одзив крана и модела позиције за x -осу

Током времена долази до „клизања” 3Д крана у једну страну, па можемо закључити да је кран „дефектан”,

односно не ради у потпуности на исправан начин по x -оси. Због тога осцилације правог крана на слици 3. нису константне амплитуде иако би требале да буду на периодичан управљачи сигнал $u(t)$ дат изразом (2). Такође, из тог разлога изгледа да пројектовани модел са естимираним параметрима не прати добро реални систем што није случај.



Слика 4. Одзив крана и модела позиције за y -осу

На основу слика 3. и 4. и објашњења везаног за x -осу можемо закључити да добијени математички модел адекватно описује кретање крана по позицији.

Метод најмањих квадрата је ткђ. кориштен за естимацију параметара a_{43} и b_4 који описују динамику угла, међутим резултати који су добијени нису били задовољавајући због превеликог утицаја сопствених осцилација. Овај проблем је ријешен тако што су параметри одређени експерименталним путем за 3 дужине: $l = 0.05m$, $l = 0.278m$ и $l = 0.5m$. Експеримент је био сљедећи: посматрана је промјена угла без дјеловања улазне силе. Наравно, тада је помјерај угла 0 rad/s . Након тога, клатно са тегом се ручно изведе из равнотежног стања што доводи до осцилација клатна. Уочено је да је овакво кретање својствено за осцилаторе другог реда и може се моделовати следећом функцијом преноса

$$G(s) = \frac{-k_\theta}{s^2 + 2\varepsilon\omega_0 s + \omega_0^2}.$$

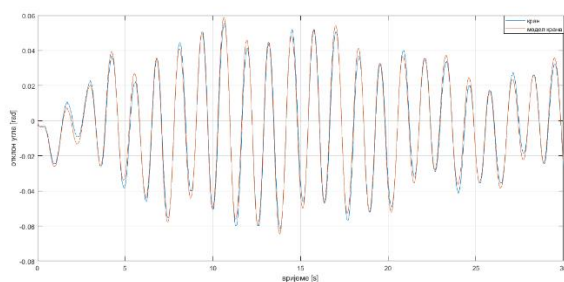
Параметар ω_0 је увијек позитиван и назива се **непригушеном природном учестаношћу**. Параметар ε је **коэффициент релативног пригушења**, а k_θ је **појачање система** и рачуна се по формули $k_\theta = \omega_0^2$.

Анализом снимљених осцилација клатна, мјерењем периоде осциловања, амплитуде и времене смирења осцилација, одређене су вриједности тражених параметара ω_0 , ε , k_θ и приказане су у Табели 2.:

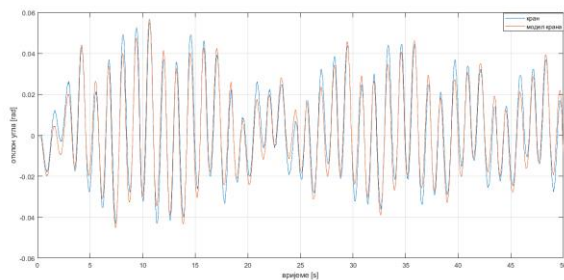
Табела 2. Добијени параметри модела за x и y -осу

	x -оса			y -оса		
l	0.005	0.278	0.5	0.005	0.278	0.5
ε	0.001	0.0025	0.03	0.001	0.0021	0.03
ω_0	5.926	4.654	3.927	5.8	4.8375	3.9
k_θ	35.124	21.661	15.423	33.64	23.401	15.21

На сликама 5. и 6. приказан је упоредни одзив крана и модела отклона угла за дужину клатна $l=0.278\text{m}$.



Слика 5. Одзив крана и модела отклона угла за x-осу



Слика 6. Одзив крана и модела отклона угла за y-осу

4. УПРАВЉАЊЕ 3Д КРАНОМ КОРИШЋЕЊЕМ ГС АЛГОРИТМА

Како се мијења динамика процеса са промјеном дужине клатна, а тиме и параметри модела процеса, за управљање краном неопходно је било примјенити адаптивни алгоритам. У овом случају одабран је ГС метод [6] који подразумијева формирање табеле параметара регулатора за различите радне тачке процеса. Управљачке структуре које су кориштене за управљање 3Д краном, а тиме и за формирање ГС скупа параметара регулатора, су регулатор по стањима, регулатор по излазима и регулатор по излазима са естимацијом поремећаја. У наставку ће бити приказани резултати за сваки од тих алгоритама, а затим ће њихови одзиви бити упоређени. Симулације су направљене тако да се референца за x и y-осу мијења током времена, а ткђ. се и дужина клатна мијења током времена. За сва 3 типа регулатора ће бити приказани одзиви за x и y-осу, промјена дужина клатна (слике 7., 9. и 11.), као и отклони углова (слике 8., 10. и 12.). ГС алгоритам је имплементиран тако да се интерполацијом рачунају параметри када се систем нађе изван радне тачке. Параметри система који су се мијењали су ω_0 , ε , k_θ . Наравно, у зависности од ових параметара, мијењају се појачања регулатора односно обсервера.

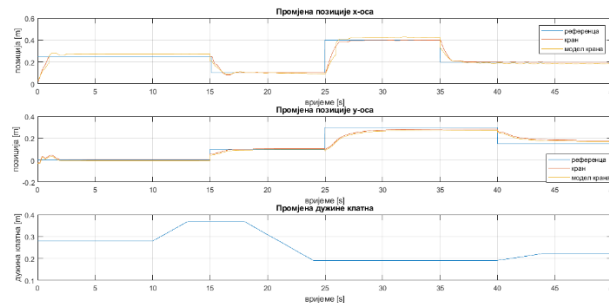
4.1. Линеарни регулатор по стањима процеса

Први управљачки закон који је имплементиран је био линеарни регулатор по стањима процеса [7].

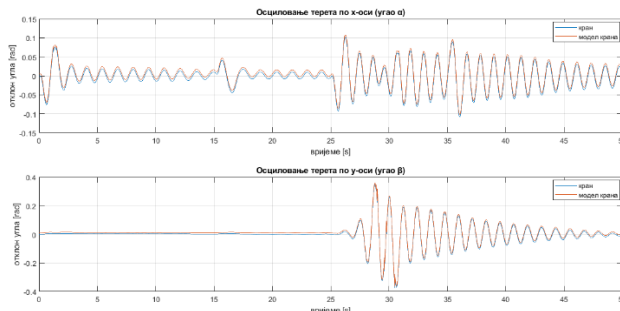
x-оса: Експерименталним путем је одређено да се најбољи резултати добију ако се полови система (1) у затвореној спреси поставе на $[-5, -4, -4, -4]$. Вриједности појачања регулатора се рачунају коришћењем Акерманове формуле, с тим што се

параметри регулатора адаптивно мијењају у зависности од дужине клатна.

y-оса: За y-осу се најбољи резултати добију ако се полови система у затвореној спреси поставе на $[-2, -1, -5, -3]$. Вриједности појачања регулатора се рачунају ткђ. по Акермановој формули.



Слика 7. Одзив крана по позицији и промјена дужине конопца-регулатор по стањима



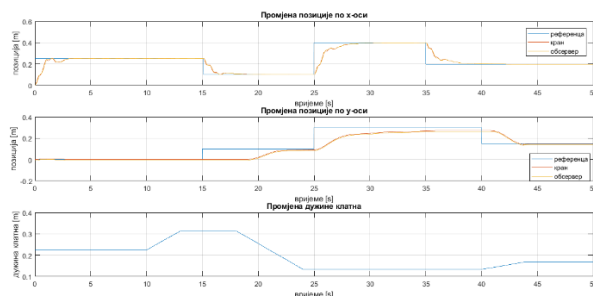
Слика 8. Отклони углова α и β терета - регулатор по стањима

4.2. Линеарни регулатор по излазима процеса

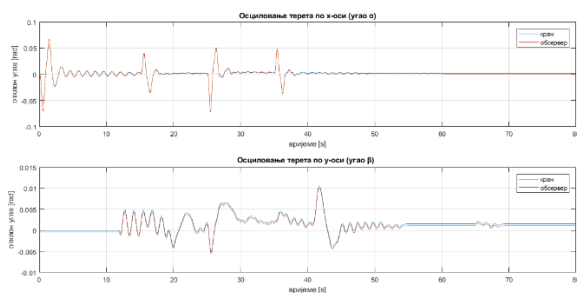
Линеарни регулатор по излазима процеса, [7], подразумијева регулатор по стањима и обсервер.

x-оса: Полови система у затвореној спреси су $[-5, -4, -4, -4]$. Полови обсервера морају бити бржи од полова система и за полове обсервера су узете вриједности $[-2\pi, -2\pi, -2\pi, -2\pi]$. Коришћењем Акерманове формуле добијамо вриједности појачања регулатора и обсервера.

y-оса: Полови система у затвореној спреси су $[-2, -1, -5, -3]$. Полови обсервера за y-осу су изабрани да буду $[-3\pi, -3\pi, -3\pi, -3\pi]$.



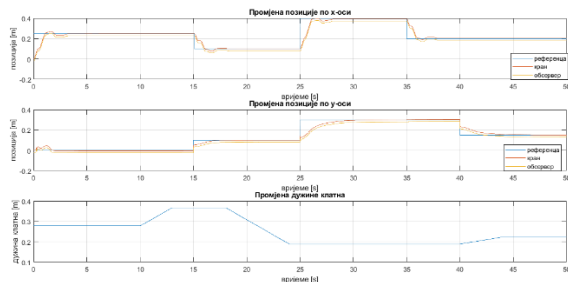
Слика 9. Одзив крана по позицији и промјена дужине конопца-регулатор по излазима



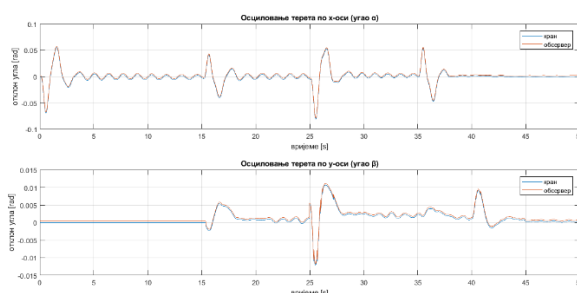
Слика 10. Отклони углова α и β терета - регулатор по излазима

4.3. Линеарни регулатор по излазима процеса са естимацијом поремећаја

Овај тип регулатора подразумева истовремену естимацију стања и поремећаја и по правилу, овај тип регулатора је знатно отпорнији на несавршености у односу на друге типове регулатора [7]. Полови у затвореној спрези, као и полови обсервера за обје осе су исти као код линеарног регулатор по излазима процеса.



Слика 11. Одзив крана по позицији и промјена дужине конопаца- регулатор по излазима са естимацијом поремећаја



Слика 12. Отклони углова α и β терета - регулатор по излазима са естимацијом поремећаја

4. ЗАКЉУЧАК

Упоредјујући одзиве ова три регулатора можемо рећи да сваки од њих даје и више него задовољавајуће резултате. Уколико посматрамо одзив крана по позицији видимо да се најбољи резултати добију када се користи линеарни регулатор по излазима процеса са естимацијом поремећаја. Коришћењем овог регулатора, кран успијева да се позиционира на тачно одређене позиције које му се задају без обзира на дужину клатна. Нешто лошији резултати се добијају линеарним регулатором по излазима, који за једну промјену позиције на у-оси има малу грешку у устаљеном стању. Што се тиче линеарног регулатора

по стањима, и он има мале грешке у устаљеном стању. Посматрајући отклоне углова долазимо до аналогног закључка: линеарни регулатор по излазима процеса са естимацијом поремећаја најбоље стабилише осциловање терета, док линеарни регулатор по стањима прави највеће осцилације. Треба нагласити да су највеће осцилације које се јављају код регулатора по стањима 0.1 rad, што је јако мало, док су углови који се јављају уколико се управља регулатором по излазима процеса са естимацијом поремећаја приближни 0 rad.

На крају можемо закључити да су се теоретске формулације ових регулатора са свим својим врлинама и манама показале довољно добрим у пракси и оправдале своју примјенљивост у управљању сложенијим системима као што је 3Д кран.

5. ЛИТЕРАТУРА

- [1] X. Wu and X. He, "Partial feedback linearization control for 3-D under actuated overhead crane systems", ISATransactions, 65, 361–370., 2016.
- [2] S.Y.S. Hussein and R. Ghazali and H.I. Jaafar, and C.C. Soon, "Analysis of 3D gantry crane system by pid and vsc for positioning trolley and oscillation reduction", Journal of Telecommunication, Electronic and Computer Engineering, 8(7), 139–143, 2016.
- [3] 3D Crane User's Manual. <http://control.put.poznan.pl/old/sites/default/files/3DCrane.pdf> (приступљено у августу 2019.)
- [4] Rudolph Emil Kalman, "A new approach to linear filtering and prediction problems", Transactions of the ASME–Journal of Basic Engineering, 82(Series D):35–45, 1960.
- [5] Мирна Капетина, "Адаптивна естимација параметара система описаних ирационалним функцијама преноса", <http://www.ftn.uns.ac.rs/n1579068189/disertacija> (приступљено у августу 2019.)
- [6] D.J. Leith, W.E. Leithead, "Gain-scheduled controller design: an analytic framework directly incorporating non-equilibrium plant dynamics", <https://pdfs.semanticscholar.org/7d45/07472fdbc2f34e09e73769b32afa11b3c26d.pdf>
- [7] Милан Р. Рапаић, Зоран Д. Јеличић., "Пројектовање линеарних регулатора и естиматора у простору стања", Факултет техничких наука, Нови Сад, 2014.

Кратка биографија:



Ања Буљевић рођена је у Београду 1996. год. Мастер рад на Факултету техничких наука из области Електротехнике и рачунарства – Аутоматика и управљање системима одбранила је 2019. год.
контакт: mail.anjabuljevic@uns.ac.rs

**FAZNO SPREGNUTA PETLJA I PROCENA NEURAVNOTEŽENOSTI MREŽE
PRIMENOM VIŠEREZONANTNOG OPSERVERA****MULTI-RESONANT OBSERVER PLL WITH ESTIMATION OF GRID UNBALANCES**

Aleksandra Mitrović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu je razmatran problem sinhronizacije na elektroenergetski sistem. Projektovan je PLL algoritam sa redukovanim estimatorom prostoperiodičnih poremećaja i RST regulatorom. Implementacija upravljačkog algoritma je izvršena u vremenski kontinualnom i diskretnom domenu.

Ključne reči: Sinhronizacija, PLL algoritam, Redukovani estimator prostoperiodičnih poremećaja, RST regulatori

Abstract – In this paper, phase locked loop or PLL algorithm for grid synchronization is investigated. The proposed technique is implemented with reduced order observer of prostoperiodic disturbances and RST controller. Algorithm control loop is implemented in continuous and discrete time domain.

Keywords: Grid synchronization, PLL algorithm, Reduced order observer, RST controller

1. UVOD

Sinhronizacija uređaja na elektroenergetski sistem je važna tema u oblasti elektroenergetike i generalno oblasti upravljanja sistemima. Uređaji koji se povezuju na elektroenergetski sistem mogu da predaju ili uzimaju energiju iz sistema u zavisnosti od radnog režima u kome se nalaze (generatorski ili motorski) a povezuju se preko savremenih uređaja energetske elektronike.

Intuitivno se može zaključiti da je prilikom priključenja uređaja na električnu mrežu važno da postoji sinhronizacija između elektroenergetskog sistema i uređaja. Ukoliko uslovi sinhronizacije nisu ispunjeni javljaju se neželjeni efekti u vidu harmonika višeg reda koji se tretiraju kao poremećaji u sistemu i neophodno ih je eliminisati. Zbog toga se projektuje fazno spregnuta petlja ili PLL algoritam kojim se vrši sinhronizacija na mrežu. U literaturi postoje različite načini za realizaciju PLL algoritma [1-3]

U okviru algoritma implementira se redukovani estimator (opserver) koji vrši procenu vrednosti nepoznatih poremećaja a potom se tako estimiran signal oduzima od merenog zašumljenog napona kako bi se dobila korisna komponenta signala. Ovakav signal se koristi u upravljačkom delu algoritma koji se implementira kao RST regulator.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bila dr Mirna Kapetina, docent.

**2. ELEKTROENERGETSKI SISTEM I
SINHRONIZACIJA SA ELEKTRIČNOM MREŽOM****2.1. Uopšteno o elektroenergetskom sistemu**

Elektroenergetski sistem obuhvata procese proizvodnje, prenosa i distribucije električne energije. Proizvodnja električne energije se vrši u elektranama (termoelektrane, hidroelektrane, nuklearne elektrane); prenos električne energije omogućava prenosna mreža (dalekovodi, kablovski vodovi, interkonektivni transformatori); distribucija je isporuka električne energije do korisnika (distributivne mreže, distributivni transformatori) [4]. Osnovni zadatak svakog elektroenergetskog sistema je da osigura kvalitetnu isporuku električne energije uz minimalne troškove.

2.2. Uslovi sinhronizacije sa električnom mrežom

Sinhronizacija je postupak kojim se uređaj dovodi u takvo radno stanje da se može priključiti na elektroenergetski sistem bez negativnih posledica. Da bi se izvršila sinhronizacija sa električnom mrežom neophodno je da se ispune sledeći uslovi: faze uređaja se moraju slagati sa fazama mreže, frekvencija uređaja mora biti jednaka frekvenciji mreže, naponi uređaja moraju biti jednaki naponima mreže i naponi uređaja moraju biti u fazi sa naponima mreže [5].

2.3. Efekti sinhronizacije sa električnom mrežom

Ukoliko neki od uslova sinhronizacije nije ispunjen dolazi do nepravilne sinhronizacije usled koje se mogu javiti velike struje izjednačavanja koje su uzrok poremećaja u mreži [5]. Ovakvi poremećaji se najčešće javljaju u vidu harmonika višeg reda. Harmonici se definišu kao neželjene spektralne komponente izobličenog signala čija je frekvencija jednaka celobrojnom umnošku osnovne frekvencije. Harmonici postoje u elektroenergetskom sistemu od početka primene naizmenične struje, međutim, njihov negativan uticaj dolazi do izražaja sa pojavom nelinearnih potražaca koji su danas brojniji u odnosu na linearne.

Shodno prethodno navedenim tvrdnjama, trofazni napon mreže u stacionarnom stanju se može predstaviti u sledećem obliku [1]:

$$u_i = U^p \cos(\omega_0 t - k_i \frac{2\pi}{3} + \varphi^p) + U^n \cos(-\omega_0 t - k_i \frac{2\pi}{3} + \varphi^n) + \sum_h U_{ih} \cos(h\omega_0 t - k_i \frac{2\pi}{3} + \varphi_{ih}) \quad (2.1)$$

gde je $h = 1 \dots n$, $k = 0, 1, -1$ i $i = a, b, c$. U^p i U^n su amplitude pozitivne i negativne sekvence osnovne frekvencije napona mreže (FFPS i FFNS) dok su φ^p i φ^n njihove faze. ω_0 je kružna učestanost napona mreže i

iznosi $\omega_0 = 2\pi \cdot 50$ rad/s. Treći sabirak u izrazu (2.1) predstavlja harmonike višeg reda koji se javljaju prilikom sinhronizacije pri čemu je U_{ih} amplituda a φ_{ih} faza h-tog harmonika.

Zbog jednostavnijeg zapisa, izraz (2.1) se može predstaviti u sledećoj formi:

$$\underline{U} = \underline{U}_1 + \sum_h \underline{U}_h \quad (2.2)$$

gde je $\underline{U}$ fazor koji odgovara naponu u i važi:

$$\begin{aligned} \underline{U}_1 &= U^p e^{j\omega_0 t} + U^n e^{-j(\omega_0 t + \varphi_n)} \\ \underline{U}_h &= U_h^p e^{j(h\omega_0 t + \varphi_h^p)} + U_h^n e^{-j(h\omega_0 t + \varphi_h^n)} \end{aligned} \quad (2.3)$$

Kako bi se uprostito račun vrši se transformacija trofaznog stacionarnog sistema (abc) u sinhrono rotirajući dvofazni sistem (dq) primenom Parkove transformacije. Ova transformacija, poznata i kao transformacija kretanja, primenjuje se kako bi se u ustaljenom stanju izbegla zavisnost kontrolisanih veličina od vremena a njena matrica transformacija je:

$$D = \begin{bmatrix} \cos(\theta_g) & \sin(\theta_g) \\ -\sin(\theta_g) & \cos(\theta_g) \end{bmatrix} \quad (2.4)$$

Nakon primene matrice transformacije na (2.3) dobija se:

$$\begin{aligned} \underline{U}^{dq} &= \underline{U} e^{-j\omega_0 t} = U^p + \\ &U^n e^{-j(2\omega_0 t + \varphi_n)} + \sum_h U_h^p e^{[j(h-1)\omega_0 t + \varphi_h^p]} + \\ &U_h^n e^{-[j(h+1)\omega_0 t + \varphi_h^n]} \end{aligned} \quad (2.5)$$

Kako je $\underline{U}^{dq} = U_d + jU_q$ sledi:

$$\begin{aligned} u_d &= U^p + U^n \cos(2\omega_0 t + \varphi_n) + \sum_h U_h^p \cos[(h-1)\omega_0 t + \varphi_h^p] + \\ &U_h^n \cos[(h+1)\omega_0 t + \varphi_h^n] + \\ u_q &= -U^n \sin(2\omega_0 t + \varphi_n) + \sum_h U_h^p \sin[(h-1)\omega_0 t + \varphi_h^p] - \\ &U_h^n \sin[(h+1)\omega_0 t + \varphi_h^n] \end{aligned} \quad (2.6)$$

Motivacija za primenu Parkove transformacije je očigledna iz izraza (2.6). Ukoliko je napon mreže u ravnoteži i bez harmonika višeg reda onda važi $u_d = U^p$ i $u_q = 0$ [3].

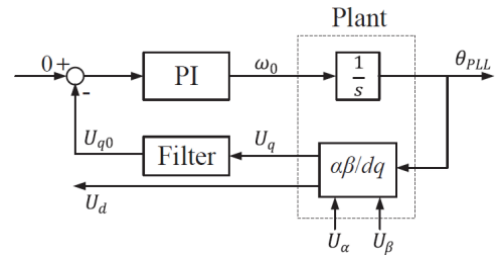
3. PLL ALGORITAM

Algoritam fazno spregnute petlje ili PLL algoritam je jedan od najzastupljenijih algoritama sinhronizacije sa električnom mrežom.

Koncept PLL algoritma je poznat od početka 20. veka i ima primenu u mnogim tehničkim disciplinama. PLL struktura je otporna na poremećaje i nesimetriju i daje brzo i tačno podatak o faznom uglu napona mreže. Upravljačka petlja algoritma prikazan je na slici 1.

PLL algoritam koristi detektor faze koji određuje razliku između faznog ugla ulazne veličine i estimiranog ugla. Dobijeni signal se filtrira i izlazni signal iz filtera pobuđuje naponski upravljani oscilator koji na svom izlazu generiše estimiranu vrednost faznog ugla koja prati fazni ugao ulazne veličine.

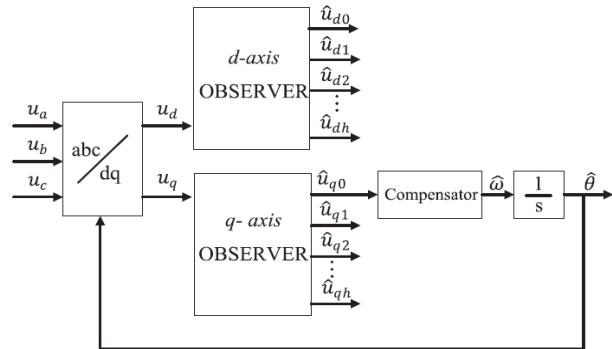
Za potrebe ovog rada posmatra se samo q osa iz sinhrono rotirajućeg referentnog sistema što se može uočiti i na slici 1.



Slika 1. Upravljačka petlja PLL algoritma [3]

4. PROJEKTOVANJE OPSERVERA REDUKOVANOG REDA

Da bi se izvršila procena vrednosti poremećaja projektovan je estimator redukovanog reda u okviru PLL petlje. Estimatori (opserveri) su algoritmi koji omogućavaju procenu vrednosti nepoznate veličine od interesa na osnovu poznavanja ulaznih i izlaznih signala [6]. U slučaju estimatora redukovanog reda, kako bi se dodatno skratila procedura projektovanja opservera, vrši se podela promenljivih na direktno merljive i nepoznate veličine. Shodno tome, upravljačka petlja PLL algoritma sa ovako implementiranim estimatorom je prikazana na slici 2 [1]:



Slika 2. PLL algoritam sa estimatorom poremećaja [3]

4.1. Vremenski kontinualni domen

Transformisani napon u_q je predstavljen izrazom (2.6). S obzirom da superpozicija bilo koja dva periodična signala iste frekvencije daje takođe periodični signal iste frekvencije, izraz (2.6) se može zapisati u sledećem obliku:

$$u_q = u_{q0} + \sum_h u_{qh}, \quad u_{qh} = U_{qhm} \sin(\omega_h t - \varphi_{qh}) \quad (4.1)$$

gde je U_{qhm} amplituda a φ_{qh} je faza h-tog harmonika u dq koordinatnom sistemu. Pošto se estimacija vrši tako što se oponaša (imitira) proces koji generiše veličinu od interesa [4], projektuje se linearni sistem sa impulsnim odzivom jednakim naponu u_q [3]:

$$\dot{u}_{q0} = 0, \quad \ddot{u}_{qh} + \omega_h^2 u_{qh} = 0 \quad (4.2)$$

Prva jednačina opisuje jednosmernu komponentnu signala dok druga opisuje harmonike višeg reda. Na osnovu navedenog matematičkog modela (4.2) može se izvršiti izbor promenljivih za h različitih harmonika:

$$x_1 = u_q, \quad x_{2h} = u_{qh}, \quad x_{2h+1} = \dot{u}_{qh} \quad (4.3)$$

odnosno:

$$\dot{x}_1 = \sum_h x_{2h+1}, \quad \dot{x}_{2h} = x_{2h+1}, \quad \dot{x}_{2h+1} = -\omega_h^2 x_{2h} \quad (4.4)$$

Nakon izbora promenljivih stanja vrši se projektovanje estimatora redukovanog reda. Promenljive stanja se dele u grupe direktno merljivih veličina x_1 i nepoznatih veličina koje se smeštaju u vektor $x_2 = [x_2 \ x_3 \ \dots \ x_{2n+1}]^T$:

$$\begin{aligned}\dot{x}_1(t) &= A_{11}x_1(t) + A_{12}x_2(t) \\ \dot{x}_2(t) &= A_{21}x_1(t) + A_{22}x_2(t)\end{aligned}\quad (4.5)$$

gde je:

$$A_{11} = 0, \ A_{12} = [0 \ 1 \ 0 \ \dots \ 1]_{2n}, \ A_{21} = [0 \ 0 \ 0 \ \dots]_{2n}^T \text{ i}$$

$$A_{22} = \begin{bmatrix} 0 & 1 & 0 & 0 & \dots \\ -\omega_1^2 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 1 & \dots \\ 0 & 0 & -\omega_2^2 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Matematički model redukovanog estimatora je:

$$\dot{z} = A_0 z + G u_q, \quad \hat{x}_2 = z + L u_q \quad (4.6)$$

Matrica L se naziva matrica pojačanja estimatora a parametarske matrice A_0 i G se računaju na sledeći način:

$$A_0 = A_{22} - L A_{12}, \quad G = A_{21} - L A_{11} + A_0 L \quad (4.7)$$

Pogodnim izborom matrice L svojstvene vrednosti matrice A_0 se smeštaju sa leve strane imaginarne ose u kompleksnoj ravni. Polovi estimatora se računaju na osnovu karakterističnog polinoma:

$$f_0(s) = s^{2n} + p_{2n-1}s^{2n-1} + \dots + p_1 s + p_0 \quad (4.8)$$

Matrica pojačanja estimatora L se računa na sledeći način:

$$f_0(s) = \det(sI - A_{22} + L A_{12}) \quad (4.9)$$

Nakon izračunavanja matrice L računaju se matrice A_0 i G . Na kraju se jednosmerna komponenta napona mreže dobija kao:

$$\hat{u}_{q0} = x_1 - \sum_h \hat{x}_{2h} \quad (4.10)$$

Veličina $\hat{u}_{q0}$ se koristi kao ulazni signal u regulacioni deo PLL petlje.

4.2. Vremenski diskretni domen

Transformisani napon u_q se u diskretnom domenu može zapisati u sledećem obliku:

$$u_q = u_{q0} + \sum_h u_{qh}, \quad u_{qh} = U_{qhm} \sin(\omega_h kT - \varphi_{qh}) = U_{qhm} \sin(\theta_h k - \varphi_{qh}) \quad (4.11)$$

gde je $\theta_h = \omega_h T$ diskretna (digitalna) kružna učestanost a T je perioda odabiranja.

Kao i u vremenski kontinualnom domenu, projektuje se linearni sistem koji ima impulsni odziv jednak naponu u_q :

$$u_{qh}(k+2) - 2 \cos(\theta_h) u_{qh}(k+1) + u_{qh}(k) = 0 \quad (4.12)$$

Na osnovu matematičkog modela vrši se izbor promenljivih stanja:

$$x_1 = u_q(k), \ x_{2h} = u_{qh}(k), \ x_{2h+1} = u_{qh}(k+1) \quad (4.13)$$

odnosno:

$$x_1(k+1) = x_1(k) - \sum_h x_{2h}(k) + \sum_h x_{2h+1}(k),$$

$$x_{2h}(k+1) = x_{2h+1}(k),$$

$$x_{2h+1}(k+1) = -x_{2h}(k) + 2 \cos(\theta_h) x_{2h+1}(k) \quad (4.14)$$

Ovako definisane promenljive stanja se dele u grupu direktno merljivih veličina x_1 i grupu nepoznatih veličina koje se smeštaju u vektor $x_2 = [x_2 \ x_3 \ \dots \ x_{2n+1}]^T$:

$$\begin{aligned}x_1(k+1) &= A_{11}x_1(k) + A_{12}x_2(k) \\ x_2(k+1) &= A_{21}x_1(k) + A_{22}x_2(k)\end{aligned}\quad (4.15)$$

gde je:

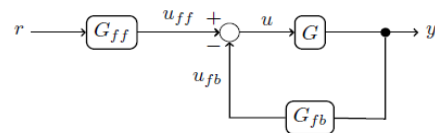
$$A_{11} = 1, \ A_{12} = [-1 \ 1 \ -1 \ \dots \ 1]_{2n}, \ A_{21} = [0 \ 0 \ 0 \ \dots]_{2n}^T \text{ i}$$

$$A_{22} = \begin{bmatrix} 0 & 1 & 0 & 0 & \dots \\ -1 & 2\cos(\theta_1) & 0 & 0 & \dots \\ 0 & 0 & 0 & 1 & \dots \\ 0 & 0 & -1 & 2\cos(\theta_2) & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Sva preostala izvođenja vezana za matematički model redukovanog estimatora, određivanje matrica L , A_0 i G i napona $\hat{u}_{q0}$ su identična kao u vremenski kontinualnom domenu, jednačine (4.6) – (4.10).

5. PROJEKTOVANJE REGULACIONOG DELA ALGORITMA U VIDU RST REGULATORA

RST regulatori predstavljaju strukturu koja je proširenje serije klasičnih regulatora jer se uvodi dodatni stepen slobode koji omogućava dizajneru da odredi ponašanje sistema u zatvorenoj povratnoj petlji na različite signale reference i poremećaja. U strukturi klasičnog RST regulatora postoje dva stepena slobode, komponenta direktnog upravljanja i komponenta upravljanja u povratnoj grani. Struktura RST regulatora je prikazana na slici 3:



Slika 3. RST regulator

gde je funkcija prenosa procesa $G(s) = \frac{B(s)}{A(s)}$, $G_{ff}(s) = \frac{T(s)}{R(s)}$ je komponenta upravljanja u direktnoj grani a $G_{fb}(s) = \frac{S(s)}{R(s)}$ je komponenta upravljanja u povratnoj grani.

5.1. Vremenski kontinualni domen

Upravljački signal na osnovu slike 3 je oblika:

$$u(t) = -u_{fb}(t) + u_{ff}(t) \quad (5.1)$$

Primenom Laplasove transformacije na izraz (5.1) i uvrštavanjem izraza za $G_{ff}(s)$ i $G_{fb}(s)$ dobija se tipična struktura RST regulatora u kompleksnom domenu:

$$RU(s) = -SY(s) + TR(s) \quad (5.2)$$

pa se na osnovu ovoga mogu dobiti izrazi za funkciju spregnutog prenosa i karakteristični polinom:

$$G_{cl}(s) = \frac{BT}{AR+BS}, \quad f_{cl}(s) = AR + BS \quad (5.3)$$

R i S su polinomi koji se računaju na sledeći način:

$$R = s f_0(s) R_1(s), \quad S = k_0 \prod_h (s^2 + \omega_h^2) S_1(s) \quad (5.4)$$

pri čemu je $k_0 = \frac{p_0}{\prod_h \omega_h^2}$ pojačanje estomatora i računa se iz uslova da je ukupno statičko pojačanje jednako nuli. Takođe, funkcija prenosa estomatora je:

$$G_0(s) = \frac{U_{q0}(s)}{U_q(s)} = k_0 \frac{\prod_h (s^2 + \omega_h^2)}{f_0(s)} \quad (5.6)$$

Na osnovu navedenog može se formirati karakteristični polinom celokupnog sistema na osnovu koga se određuju nepoznati parametri:

$$f_c(s) = s^2 f_0(s) R_1(s) + k_0 k_T S_1(s) \prod_h (s^2 + \omega_h^2) \quad (5.7)$$

pri čemu je k_T pojačanje dq transformacije nakon linearizacije. Da bi prethodna jednačina bila rešiva neophodno je da broj promenljivih bude jednak stepenu sistema pa sledi:

$$R_1(s) = 1, \quad S_1(s) = k_p(s + \sigma)$$

Nepoznati parametri se dobijaju metodom podešavanja polova. Prvo je potrebno definisati željeni karakteristični polinom koga određuju polovi koji definišu ponašanje sistema a potom se koeficijenti tako definisanog polinoma izjednačavaju sa koeficijentima iz $f_c(s)$.

5.2. Vremenski diskretni domen

Sva izvođenja i postupak određivanja nepoznatih parametara navedeni u vremenski kontinualnom domenu važe i u vremenski diskretnom domenu. Shodno tome, u nastavku će biti samo navedeni krajnji izrazi od interesa.

Funkcija prenosa estomatora u diskretnom domenu je:

$$G_0(z) = \frac{U_{q0}(z)}{U_q(z)} = k_0 \frac{\prod_h (z^2 - 2z \cos(\theta_h) + 1)}{f_0(z)} \quad (5.8)$$

gde je pojačanje estomatora $k_0 = \frac{p_0}{\prod_h (2 - 2 \cos(\theta_h))}$, karakteristični polinom estomatora je $f_0(z) = \det(zI - A_{22} + LA_{12})$ a karakteristični polinom celokupnog sistema je:

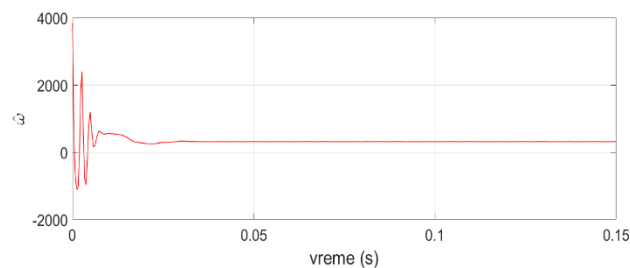
$$f_c(z) = (z - 1)^2 f_0(z) R_1(z) + k_0 k_T S_1(z) \prod_h (z^2 - 2z \cos(\theta_h) + 1) \quad (5.9)$$

pri čemu je $R_1(z) = 1$ i $S_1(z) = k_p(z + \sigma)$.

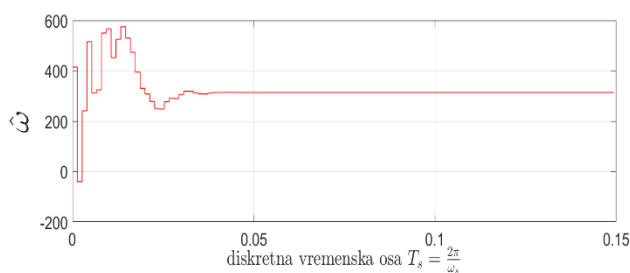
6. EKSPERIMENTALNI REZULTATI

Predloženi PLL algoritam primenjen je da bi se kroz simulacije testirala sinhronizacija mreže uz prisustvo viših harmonika u naponu mreže.

Razmatra se slučaj kada postoje dva poremećaja na frekvencijama $\omega_1 = 2\omega_0$ i $\omega_2 = 3\omega_0$ gde je $\omega_0 = 2 \cdot 50\pi$ rad/s. Dominantni polovi RST regulatora u PLL algoritmu se biraju tako da imaju prirodnu neprigušenu učestanost ω_0 i relativni koeficijent prigušenja $\xi = 0.7$ dok se ostali polovi nalaze dalje od imaginarne ose. Opisan algoritam je testiran i u vremenski kontinualnom i diskretnom domenu kada se primenjuje perioda odabiranja je $T_s = \frac{2\pi}{\omega_s}$ gde je $\omega_s = 5\omega_2$. Dobijeni rezultati su prikazani na likama 4 i 5. Sa slike 4 se jasno vidi da je algoritam realizovan u kontinualnom domenu uspešno izdvojio iz „zagadenog” signala osnovni harmonik na frekvenciji $\omega_0 = 2 \cdot 50\pi$ rad/s, a slika 5 prikazuje rezultate simulacije predloženog algoritma realizovanog u diskretnom domenu.



Slika 4. Estimirana kružna učestanost napona mreže



Slika 5. Estimirana kružna učestanost napona mreže

7. LITERATURA

- [1] M. Vekić, M. R. Rapaić, T. B. Šekara, S. Grabić, E. Adžić, „Multi-resonant observe PLL with real-time estimation of grid unbalances”, International Journal of Electrical Power & Energy Systems, Vol. 108, pages 52-60, June 2019.
- [2] H. Awad, J. Svensson and M. J. Bollen, „Tuning software phase-locked loop for series-connected converters,” in IEEE Transactions on Power Delivery, vol. 20, no. 1, pp. 300-308, Jan. 2005. doi: 10.1109/TPWRD.2004.837823
- [3] W. Li, X. Ruan, C. Bao, D. Pan and X. Wang, „Grid Synchronization Systems of Three-Phase Grid-Connected Power Converters: A Complex-Vector-Filter Perspective,” in IEEE Transactions on Industrial Electronics, vol. 61, no. 4, pp. 1855-1870, April 2014.
- [4] S. Branković, „Električne mreže i dalekovodi”, Zavod za udžbenike i nastavna sredstva, Beograd, 1996.
- [5] M. Božić, M. Rosić, M. Bjekić, „Vizuelizacija postupka sinhronizacije sinhronog generatora na mrežu”, 6. Međunarodni simpozijum, Tehnički fakultet Čačak, 3-5. Jun 2011.
- [6] M. R. Rapaić, Z. D. Jeličić, „Projektovanje linearnih regulatora i estomatora u prostoru stanja”, Univerzitet u Novom Sadu, Fakultet tehničkih nauka, 2014.

Kratka biografija



Aleksandra Mitrović rođena je u Zvorniku 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Adaptivno i napredno upravljanje odbranila je 2019.god.

Kontakt: ana.mitroviczv@gmail.com

**HIBRIDNI ESTIMATOR STANJA SA STRUJAMA U PRAVOUGAONIM
KOORDINATAMA ZA UKLJUČIVANJE FAZORSKIH MERENJA****HYBRID POWER SYSTEM STATE ESTIMATION WITH PHASOR-BASED CURRENT
MEASUREMENTS IN RECTANGULAR COORDINATES**

Jovana Kovačević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu je obrađen problem hibridne estimacije stanja, koja uključuje kako konvencionalna merenja, tako i sinhronizovana merenja fazora. Metod minimuma sume otežanih kvadrata reziduala merenja (WLS metod) uvažava strujna merenja u pravougaonim koordinatama.

Ključne reči: Estimacija stanja, Hibridni estimator stanja, Jedinica za merenje fazora (PMU), Sinhronizovana merenja fazora

Abstract – In this paper the hybrid state estimation based on conventional and synchronized phasor measurements is analyzed. Current measurements in rectangular coordinates are included to the Weighted Least Squares (WLS) state estimation.

Keywords: State estimation, Hybrid power system state estimation, Phasor Measurement Unit (PMU), Synchronized phasor measurements

1. UVOD

Proračun estimacije stanja elektroenergetskog sistema (EES-a) predstavlja ključni proračun u savremenim upravljačkim centrima. Zadatak estimatora stanja jeste određivanje nepoznatih vrednosti promenljivih stanja na bazi redundantnih merenja iz sistema, gde se rezultati dobijeni ovim proračunom dalje koriste za proračun drugih sistemskih promenljivih od interesa. U svrhu procene stanja EES-a u većini slučajeva se koristi metod minimuma sume otežanih (ponderisanih) kvadrata reziduala merenja, poznatiji kao WLS (engl. Weighted Least Square) metod [1]. Cilj statičke estimacije stanja jeste najbolja moguća procena stanja EES-a u datom trenutku, koja će biti efikasna i pouzdana, a da pritom uvaži šumove pri merenjima, gubitak merenja, pogrešne parametre i slično.

Pod pojmom klasičnog estimatora stanja podrazumeva se estimator čije ulazne podatke čine konvencionalna merenja dostupna korišćenjem SCADA (engl. Supervisory Control and Data Acquisition) sistema. U okviru master rada objašnjeni su osnovni pojmovi u estimaciji stanja: vektor promenljivih stanja, vektor merenja, redundansa merenja, težinski faktori merenja. Takođe, opisan je algoritam WLS estimacije stanja, sa po-

sebnim osvrtom na problem slabe uslovljenosti promenljivih.

Kako bi li se postigla veća sigurnost i efikasnije vođenje savremenog EES-a, sinhronizovana merenja fazora (u literaturi su u upotrebi i sinonimi: PMU merenja i fazorska merenja) su tehnologija koja je prepoznata kao jedno od ključnih rešenja.

2. PMU MERENJA

PMU merenja se odlikuju sledećim karakteristikama [2]:

- velikom učestanošću merenja, koja se kreće i do 60 merenja u sekundi (tipični PMU uređaji generišu 48 merenja u sekundi);
- velikom tačnošću merenja;
- većom zaštićenošću od sajber napada u odnosu na konvencionalna merenja;
- značajno većom cenom u odnosu na klasična SCADA merenja.

Ugradnjom PMU uređaja u određene čvorove EES-a postaju dostupna sinhronizovana merenja fazora napona čvorova, kao i fazori struja duž grana koje su koincidentne čvorovima ugradnje. Takođe, PMU jedinica izračunava frekvenciju u sistemu, kao i brzinu promene frekvencije. Osnovna prednost PMU merenja u poređenju sa konvencionalnim merenjima dobijenim iz SCADA sistema jeste merenje uglova napona i struja u EES-u, kao i mogućnost usklađivanja merenja iz geografski udaljenih PMU jedinica – što omogućuje vremenska oznaka, nezavisno od vremena dolaska u upravljački centar.

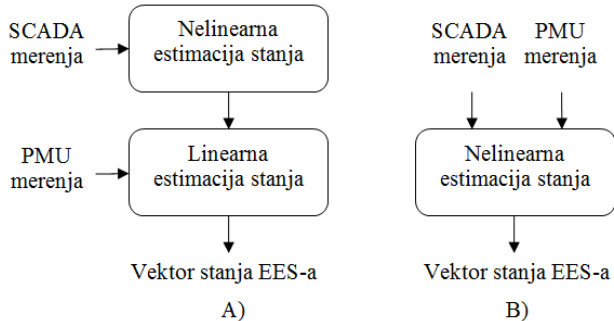
3. HIBRIDNI ESTIMATOR STANJA SA STRUJAMA U PRAVOUGAONIM KOORDINATAMA

U zavisnosti od arhitekture hibridnog modela estimatora, postoji podela na dve kategorije (kao što je prikazano na slici 1). Varijanta A) prikazuje hibridni algoritam u kom je konvencionalna merenja moguće koristiti u klasičnom nelinearnom postupku (pri čemu je EES opservabilan samo korišćenjem SCADA merenja). Rezultat klasičnog estimatora stanja predstavlja estimirani vektor stanja svih čvorova u EES-u, koji se zatim zajedno sa PMU merenjima koristi u linearnom estimatoru stanja. Osnovna mana ovakvog pristupa jeste to što dodatno uključivanje PMU merenja ne doprinosi opservabilnosti sistema, a samim tim ni detekciji i identifikaciji grubih grešaka. Varijanta B) prikazuje hibridni estimator stanja koji koristi vektor merenja sastavljen od SCADA merenja i PMU merenja, što zahteva modifikaciju algoritma u upravljačkom centru i promene u menadžment sistemu

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Andrija Sarić, red. prof.

prenosne mreže. Međutim, prednost ovog pristupa predstavlja korišćenje fazorskih zajedno sa SCADA merenjima, čime se otklanja navedena mana prvog pristupa, a istovremeno doprinosi redundansi merenja u EES-u. U ovom radu obrađena je upravo ovakva vrsta pristupa, gde se SCADA i PMU merenja kombinuju u nelinearnom estimatoru stanja.



Slika 1. Arhitekture postojećih hibridnih modela estimatora stanja [3]

3.1. Kombinovanje konvencionalnih i PMU merenja

Korišćenjem dve vrste vremenskih oznaka moguće je kombinovati sinhronizovana fazorska merenja i konvencionalna merenja, koja nisu sinhronizovana. U slučaju da nema trenutnog podudaranja među merenjima, jedna vrsta merenja može biti prevedena u drugu interpolacijom. Uobičajena praksa je da se za referentni fazni ugao uzima fazni ugao napona merenog u balansnom čvoru mreže. Bez gubitka opštosti, pretpostavlja se da je PMU uređaj instaliran u balansnom čvoru. Ostali fazni uglovi, mereni ili estimirani, za referentni fazni ugao uzimaju fazni ugao napona balansnog čvora [4].

3.2. PMU merenja u estimaciji stanja EES-a

Ukoliko bi se PMU uređaj instalirao u svakom od čvorova EES-a, opservabilnost celog sistema ostvarila bi se isključivo pomoću fazorskih merenja. Očigledno je da bi u ovom slučaju proračun estimacije stanja bio nepotreban, s obzirom da se promenljive stanja na ovaj način mere direktno. Stoga, veza merenja i promenljivih stanja je linearna, a samim tim ne postoji potreba za iterativnim postupkom i estimacijom stanja kao takvom. Osim toga što ovo nije realna opcija, s obzirom na velike cene PMU uređaja, ovaj pristup nije dobar ni iz tehničkih razloga. Naime, ako raspored PMU jedinica ne osigurava redundantnost merenja, u slučaju gubitka ili netačnosti fazorskih merenja (otkaz PMU jedinice, greška mernog transformatora, komunikacijska greška) dolazi do degradacije rezultata estimacije, uz nemogućnost detekcije i identifikacije grubih grešaka u merenju i grešaka u modelu mreže [3]. Potpunu opservabilnost sistema moguće je postići kombinovanjem konvencionalnih i fazorskih merenja, što zahteva ugradnju PMU uređaja samo u delu čvorova EES-a. Ovo je finansijski znatno isplativiji i, samim tim, dosta korišćeniji pristup s obzirom na brojnost već postojećih mernih uređaja koji čine sastavni deo SCADA sistema, čime osiguravaju redundantnost merenja. Uvođenje PMU merenja u proračun estimacije stanja ima pozitivan uticaj na ostale proračune i analize u upravljačkom centru, što

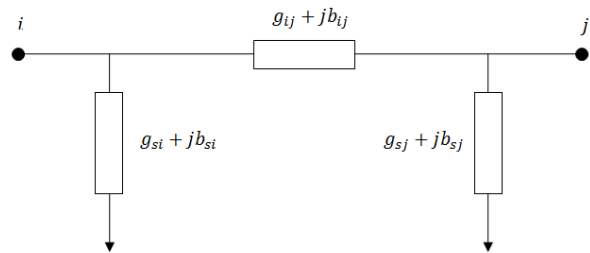
za rezultat ima veći stepen raspoloživosti EES-a i sigurnosti pogona EES-a u stvarnom vremenu.

3.3. Formulacija hibridnog estimatora stanja

Fazor struje $\bar{I}_{ij} = I_{ij} \angle \theta_{ij}$, koja teče od čvora i prema čvoru j , može primenom Prvog Kirhofovog zakona za čvor i biti izražen sledećim izrazom [1]:

$$\bar{I}_{ij} = \bar{V}_i(g_{si} + jb_{si}) + (\bar{V}_i - \bar{V}_j)(g_{ij} + jb_{ij}), \quad (1)$$

gde su fazori $\bar{V}_i = V_i \angle \theta_i$ i $\bar{V}_j = V_j \angle \theta_j$ naponi u čvorovima i i j , respektivno; dok $g_{si} + jb_{si}$ predstavlja otočnu admitansu u čvoru i , a $g_{ij} + jb_{ij}$ rednu admitansu između čvorova i i j (kao što je prikazano na slici 2, gde je sa $g_{sj} + jb_{sj}$ označena otočna admitansu u čvoru j).



Slika 2. π -model voda [1]

Direktna primena fazora struje predstavljenog preko polarnih koordinata u estimaciji stanja je moguća, pri čemu je neophodno koristiti WLS iterativni metod – s obzirom na to da je vektor stanja sastavljen od fazora napona čvorova u polarnim koordinatama, što za posledicu ima nelinearnu vezu s pojedinim elementima vektora merenja, kojeg čine konvencionalna merenja i fazori napona i struja u polarnim koordinatama. Usled slabe uslovljenosti matrice pojačanja, sa primenom ovog metoda pojavljuju se problemi sa konvergencijom proračuna. Stoga, pristup gde se fazori struja uvažavaju preko polarnih koordinata se ne preporučuje i dalje neće biti razmatran. Ovakva vrsta problema nije uočena pri primeni metoda koji koristi fazore struja predstavljene pomoću pravougaonih koordinata.

3.3.1. Vektor merenja

Vektor merenja čine konvencionalna merenja, fazori napona u polarnim koordinatama (direktno se uzimaju merenja amplitude napona i ugla napona u čvoru), kao i fazori struja u pravougaonim koordinatama:

$$\mathbf{z} = [\mathbf{P}^T \mathbf{Q}^T \mathbf{P}_l^T \mathbf{Q}_l^T \boldsymbol{\theta}^T \mathbf{V}^T \mathbf{I}_{l,r}^T \mathbf{I}_{l,i}^T]^T, \quad (2)$$

gde su:

- $\mathbf{P}$ – vektor injektiranja aktivnih snaga u čvorovima (konvencionalna merenja);
- $\mathbf{Q}$ – vektor injektiranja reaktivnih snaga u čvorovima (konvencionalna merenja);
- $\mathbf{P}_l$ – vektor tokova aktivnih snaga po granama (konvencionalna merenja);
- $\mathbf{Q}_l$ – vektor tokova reaktivnih snaga po granama (konvencionalna merenja);
- $\boldsymbol{\theta}$ – vektor uglova napona u čvorovima (direktno merenja promenljive stanja, PMU merenja);
- $\mathbf{V}$ – vektor modula napona u čvorovima (direktno merenja promenljive stanja, kombinacija konvencionalnih i PMU merenja);

$I_{L,R}$ – vektor realnih komponenti tokova struja po granama (PMU merenja);

$I_{L,I}$ – vektor imaginarnih komponenti tokova struja po granama (PMU merenja).

S obzirom na nelinearnu vezu kako konvencionalnih merenja, tako i merenja struje s elementima vektora stanja, jasno je da će hibridni estimator stanja sa strujama u pravougaonim koordinatama koristiti iterativnu WLS metodu.

3.3.2. Jakobijana matrica

Elementi matrice Jakobijana, kao i u klasičnom estimatoru stanja, dobijaju se parcijalnim izvodima po elementima vektora stanja, kojeg čine fazori napona čvorova u polarnim koordinatama. S obzirom da se fazori napona uvažavaju u polarnim, a fazori struja u pravougaonim koordinatama, elementi matrice Jakobijana predstavljeni su na sledeći način:

$$H(x) = \begin{bmatrix} \frac{\partial P(x)}{\partial \theta} & \frac{\partial P(x)}{\partial V} \\ \frac{\partial Q(x)}{\partial \theta} & \frac{\partial Q(x)}{\partial V} \\ \frac{\partial P_I(x)}{\partial \theta} & \frac{\partial P_I(x)}{\partial V} \\ \frac{\partial Q_I(x)}{\partial \theta} & \frac{\partial Q_I(x)}{\partial V} \\ \frac{\partial \theta(x)}{\partial \theta} & \frac{\partial \theta(x)}{\partial V} \\ \frac{\partial V(x)}{\partial \theta} & \frac{\partial V(x)}{\partial V} \\ \frac{\partial I_{L,R}(x)}{\partial \theta} & \frac{\partial I_{L,R}(x)}{\partial V} \\ \frac{\partial I_{L,I}(x)}{\partial \theta} & \frac{\partial I_{L,I}(x)}{\partial V} \end{bmatrix}. \quad (3)$$

Elementi submatrica matrice Jakobijana koji se odnose na SCADA merenja ($\frac{\partial P(x)}{\partial \theta}$, $\frac{\partial P(x)}{\partial V}$, $\frac{\partial Q(x)}{\partial \theta}$, $\frac{\partial Q(x)}{\partial V}$, $\frac{\partial P_I(x)}{\partial \theta}$, $\frac{\partial P_I(x)}{\partial V}$, $\frac{\partial Q_I(x)}{\partial \theta}$, $\frac{\partial Q_I(x)}{\partial V}$) mogu se dobiti standardnim postupkom diferenciranja jednačina za tokove snaga. Submatrice $\frac{\partial \theta(x)}{\partial \theta}$ i $\frac{\partial V(x)}{\partial V}$ su jedinične matrice, koje se formiraju tako što se samo na mestima na kojima se merenje ugla, odnosno amplitude napona u čvoru i podudara sa promenljivom stanja u čvoru i upisuje jedinica u koloni i odgovarajuće vrste, dok su svi ostali elementi nule. Svi elementi submatrica $\frac{\partial \theta(x)}{\partial \theta}$ i $\frac{\partial V(x)}{\partial V}$ su nule.

Elementi submatrica $\frac{\partial I_{L,R}(x)}{\partial \theta}$, $\frac{\partial I_{L,R}(x)}{\partial V}$, $\frac{\partial I_{L,I}(x)}{\partial \theta}$ i $\frac{\partial I_{L,I}(x)}{\partial V}$ su [1]:

$$\frac{\partial I_{ij,R}}{\partial \theta_i} = -V_i [\sin \theta_i (g_{si} + g_{ij}) + \cos \theta_i (b_{si} + b_{ij})], \quad (4)$$

$$\frac{\partial I_{ij,R}}{\partial \theta_j} = V_j (b_{ij} \cos \theta_j + g_{ij} \sin \theta_j), \quad (5)$$

$$\frac{\partial I_{ij,R}}{\partial V_i} = \cos \theta_i (g_{si} + g_{ij}) - \sin \theta_i (b_{si} + b_{ij}), \quad (6)$$

$$\frac{\partial I_{ij,R}}{\partial V_j} = b_{ij} \sin \theta_j - g_{ij} \cos \theta_j, \quad (7)$$

$$\frac{\partial I_{ij,I}}{\partial \theta_i} = -V_i [\sin \theta_i (b_{si} + b_{ij}) - \cos \theta_i (g_{si} + g_{ij})], \quad (8)$$

$$\frac{\partial I_{ij,I}}{\partial \theta_j} = V_j (b_{ij} \sin \theta_j - g_{ij} \cos \theta_j), \quad (9)$$

$$\frac{\partial I_{ij,I}}{\partial V_i} = \sin \theta_i (g_{si} + g_{ij}) + \cos \theta_i (b_{si} + b_{ij}), \quad (10)$$

$$\frac{\partial I_{ij,I}}{\partial V_j} = -b_{ij} \cos \theta_j - g_{ij} \sin \theta_j. \quad (11)$$

3.3.3. Nesigurnosti merenja

U slučaju hibridnog estimatora stanja sa strujama u pravougaonim koordinatama potrebne su modifikacije određenih elemenata matrice kovarijansi merenja (R), koja u ovom slučaju više nije dijagonalna matrica.

Najpre je potrebno navesti izraz čijim se kvadriranjem dobijaju varijanse merenja [1]:

$$\sigma_i = \frac{\Delta \sigma_i}{\sqrt{3}}, \quad (12)$$

gde je $\Delta \sigma_i$ maksimalna standardna devijacija i -tog merenja, uz pretpostavku da se radi o normalnoj raspodeli na celom rasponu nesigurnosti merenja. Maksimalne standardne devijacije merenja, koje služe za određivanje varijansi merenja, prikazane su u tabeli 1.

Tabela 1. Maksimalne standardne devijacije merenja [3]

Konvencionalna (SCADA) merenja	
Snage injektiranja	2%
Tokovi snaga	2%
Amplituda napona	0.2%
Sinhronizovana fazorska merenja	
Amplituda fazora struje	0.03%
Amplituda fazora napona	0.02%
Fazni ugao	0.01°

Polazeći od pretpostavke da su sva merenja koja učestvuju u hibridnom estimatoru stanja [prikazana izrazom (2)] međusobno nezavisna, vandijagonalni elementi u matrici kovarijansi merenja (R) su nule. Izuzetak od ovog pravila predstavljaju isključivo merenja faznog ugla, odnosno njima odgovarajući vandijagonalni elementi u matrici kovarijansi merenja (R), iz razloga koji će u nastavku biti objašnjen. Dakle, za sva ostala merenja odgovarajući dijagonalni elementi matrice kovarijansi predstavljeni su izrazom [1]:

$$R(i, i) = \sigma_i^2, \quad (13)$$

dok su vandijagonalni elementi nule, osim u onim vrstama koje se odnose na merenja faznog ugla. S obzirom da se radi o metodi u kojoj su jedino naponi izraženi u polarnim koordinatama, isključivo u vrstama matrice kovarijansi koje odgovaraju merenjima faznog ugla napona postoje nenulti vandijagonalni elementi i predstavljeni su sledećim izrazom [1]:

$$R(i, j) = \sigma_0^2, \quad (14)$$

gde je σ_0^2 varijansa merenja faznog ugla u balansnom čvoru mreže. Naime, kao što je već spomenuto, sva merenja faznog ugla sprovode se u odnosu na fazni ugao meren PMU jedinicom u balansnom čvoru mreže. Dijagonalni elementi u matrici kovarijansi koji odgovaraju merenjima faznih uglova dati su izrazom [1]:

$$R(i, i) = \sigma_i^2 + \sigma_0^2, \quad (15)$$

pa je na ovaj način uzeta u obzir nesigurnost merenja faznog ugla u ostalim čvorovima, s obzirom na nesigurnost merenja faznog ugla u referentnom čvoru.

U zavisnosti od primenjene metode, primena PMU merenja u hibridnom estimatoru stanja zahteva određene transformacije merenja. Konkretno, u slučaju metode koja se proučava u ovom radu, reč je transformaciji fazora struja iz polarnih u pravougaone koordinate. Time se unosi povećanje nesigurnosti merenja, pa je shodno tome potrebno uvažiti nove vrednosti varijansi takvog merenja, jer su one neophodne za procenu težinskih faktora u WLS metodi. Varijanse merenja realne i imaginarne komponente strujnih merenja, respektivno, mogu se dobiti kvadriranjem sledećih izraza, koji su posledica primene klasične teorije propagacije nesigurnosti merenja [1]:

$$\sigma_{I_{ij,R}} = \sqrt{\left(\frac{\partial I_{ij,R}}{\partial I_{ij}}\right)^2 \sigma_{I_{ij}}^2 + \left(\frac{\partial I_{ij,R}}{\partial \theta_{ij}}\right)^2 \sigma_{\theta_{ij}}^2}, \quad (16)$$

$$\sigma_{I_{ij,I}} = \sqrt{\left(\frac{\partial I_{ij,I}}{\partial I_{ij}}\right)^2 \sigma_{I_{ij}}^2 + \left(\frac{\partial I_{ij,I}}{\partial \theta_{ij}}\right)^2 \sigma_{\theta_{ij}}^2}. \quad (17)$$

4. VERIFIKACIJA PRORAČUNA

Verifikacija proračuna izvršena je na IEEE 14 test mreži (opis mreže i različite konfiguracije merenja nalaze se u master radu). Rezultati proračuna prikazani su u tabeli 2.

Tabela 2. Estimirani naponi u polarnim koordinatama za test sistem od 14 čvorova

Čvor	Amplituda napona [r.j.]	Ugao napona [°]
1	1.06671	0.00000
2	1.04990	-5.48096
3	1.00529	-13.80275
4	1.02200	-10.54973
5	1.02700	-9.10643
6	1.08071	-14.37381
7	1.05865	-12.45693
8	1.10226	-12.86949
9	1.04696	-14.25783
10	1.04643	-14.62403
11	1.06262	-14.71309
12	1.07332	-15.40988
13	1.06751	-15.37686
14	1.04061	-15.61007

Prikazani rezultati dobijaju se nakon 12 iteracija, gde je uslov konvergencije bio $\xi = 10^{-5}$. Ukoliko bi se promenio uslov konvergencije, tako da se usvoji da je $\xi = 10^{-4}$, rezultati proračuna ne bi značajno odstupali od već izloženih, pri čemu bi proračun konvergirao nakon 9 iteracija.

Za test mrežu od 14 čvorova u master radu su dodatno diskutovni slučajevi kada se menja skup strujnih merenja koji učestvuju u hibridnom estimatoru stanja. Dakle,

pored strujnih merenja iz originalne konfiguracije merenja koriste se dodatna strujna merenja. Uslov konvergencije u ovim slučajevima je $\xi = 10^{-4}$. Ukoliko se uključe sva raspoloživa merenja fazora struja, u slučaju ovog test sistema potrebno je 6 iteracija da bi proračun konvergirao, ali ista brzina konvergencije može se dobiti i uključivanjem samo jednog dodatnog merenja.

5. ZAKLJUČAK

U radu je prikazan hibridni algoritam za estimaciju stanja, koji se temelji na principima estimacije stanja podržane samo od strane SCADA sistema, ali su u nelinearnom estimatoru stanja uvažena i PMU merenja napona u polarnim koordinatama, kao i PMU merenja struja u pravougaonim koordinatama.

Na obrađenom primeru uočena je važnost izbora strujnih merenja koja će se koristiti u proračunu. Naime, od izbora čvora u kome će PMU uređaj biti postavljen, zavisi opservabilnost EES-a, a potom i konvergencija proračuna estimacije stanja, odnosno brzina samog proračuna. Raspored strujnih merenja koja ulaze u proračun je važniji od njihovog broja, kada je reč o broju iteracija potrebnih za konvergenciju proračuna.

Bitno je napomenuti da kvalitet merenja ima veliki uticaj na tačnost reziduala dobijenih WLS metodom, pa je poseban akcenat rada stavljen na proračun i uvažavanje varijansi različitih vrsta merenja.

Primena PMU merenja u okviru proračuna estimacije stanja predstavlja jedan od ključnih faktora u izgradnji naprednih prenosnih elektroenergetskih mreža, a njihovo kombinovanje sa konvencionalnim merenjima postaje imperativ usled toga što takvo rešenje omogućava optimalno iskorišćavanje postojećih resursa.

6. LITERATURA

- [1] S. Chakrabarti, E. Kyriakides, G. Ledwich, A. Ghosh, "Inclusion of PMU current phasor measurements in power system state estimator", *IET Generation, Transmission & Distribution*, vol. 4, no. 10, pp. 1104-1115, October 2010.
- [2] N. Živković, "Detekcija malicioznih napada na elektroenergetski sistem korišćenjem sinergije statičkog i dinamičkog estimatora", doktorski rad, Fakultet tehničkih nauka Novi Sad, 2018.
- [3] V. Kirinić, "Hibridni algoritam estimacije stanja elektroenergetskog sustava podržan sinkroniziranim mjerenjima", doktorski rad, Fakultet elektrotehnike i računarstva, Zagreb, 2013.
- [4] S. Chakrabarti, E. Kyriakides, G. Velderde, V. Terzija, "State estimation including synchronized measurements", *IEEE PowerTech*, Bucharest, Romania, 2009.

Kratka biografija:



Jovana Kovačević rođena je u Beogradu 1994. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnika i računarstvo – Elektroenergetski sistemi odbranila je 2017. god. Iste godine upisala je master studije.

PROTOKOL JEZIČKIH SERVERA LANGUAGE SERVER PROTOCOL

Bojan Stipić, Zorica Suvajdžin Rakić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Cilj ovog rada jeste analiziranje protokola jezičkih servera. Prikazana je istorija i motivacija za razvoj protokola jezičkih servera, kao i sami tehnički detalji. Opisan je programski jezik miniC. Implementiran je jezički server za programski jezik miniC. Prikazana je integracija jezičkih servera sa tekst editorima Vim i Atom.

Ključne reči: Protokol jezičkih servera, LSP, miniC

Abstract – Goal of this thesis is to analyze the Language Server Protocol. The history and motivation for developing Language Server Protocol as well as the technical details themselves are presented. The miniC programming language is described. A language server for the miniC programming language is implemented. The integration of language servers with Vim and Atom text editors is presented.

Keywords: Language Server Protocol, LSP, miniC

UVOD

Osnovni alat svakog programera je tekst editor. Tekst editori poseduju brojne funkcionalnosti koje olakšavaju život programera. Neke od često dostupnih funkcionalnosti su: dijagnostike (*diagnostics*), dovršavanje reči (*word completion*), isticanje sintakse (*syntax highlighting*), skok na definiciju (*go to definition*), pronalaženje referenci (*find references*)...

Različiti korisnici imaju različite potrebe pri korišćenju tekst editora, pa stoga i potreba za velikim brojem istih. Takođe, pored velikog broja tekst editora, postoji i veliki broj različitih programskih jezika. Implementacija gore navedenih funkcionalnosti zavisi od programskog jezika koji se koristi. Prema tome, svaki tekst editor je tradicionalno morao da implementira sve funkcionalnosti za svaki programski jezik posebno.

Bilo bi pogodnije kada bi postojali zasebni jezički alati koji pružaju "pametne" funkcionalnosti za pojedinačne programske jezike. Tada bi tekst editori mogli da komuniciraju sa tim alatima i tako pruže podršku za sve programske jezike.

Upravo to i jeste ideja Protokola jezičkih servera (Language server protocol) [1] [2] [skraćeno LSP]. LSP ima za cilj da definiše standardizovan način komunikacije između jezičkih alata (servera) i samih tekst editora.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bila dr Zorica Suvajdžin Rakić.

Na ovaj način problem $m \times n$ kompleksnosti svodi se na $m+n$ kompleksnost.

1. PROTOKOL JEZIČKIH SERVERA

1.1 Istorija i motivacija

LSP je kreiran od strane Microsoft-a kako bi se definisao zajednički jezik kojim jezički alati komuniciraju. Nastao je 2016. godine, a pored Microsofta su se udružile i druge kompanije, kao što su Red Hat, Codenvy i Sourcegraph, kako bi podržali njegov rast.

Inicijalno, Microsoft je razvio OmniSharp projekat sa ciljem da pruži "pametne" funkcionalnosti za programski jezik C# u tekst editorima. OmniSharp je inicijalno koristio HTTP protokol za razmenu JSON poruka. Ubrzo je integrisan u nekoliko tekst editora, a jedan od njih bio je VS Code, koji je takođe razvijen od strane Microsoft-a.

U istom vremenskom periodu, Microsoft je počeo sa razvojem još jednog jezičkog servera. Ovaj jezički server je imao za cilj da podrži programski jezik TypeScript. Koristio je standardni ulaz/izlaz kao transportni protokol, a poruke su bile u JSON formatu po uzoru na V8 debugger protokol.

Kada je tim zadužen za razvoj VS Code editora integrisao i TypeScript jezički server, uvideli su da je konzumacija jezičkih servera suviše komplikovana. Kako bi pojednostavili integraciju, počeli su da razmišljaju o opštem zajedničkom protokolu [3]. Zajednički protokol omogućio bi da se integracioni kod napiše samo jednom, i da se koristi za sve jezičke servere koji pričaju zajednički jezik.

Zajednički protokol je kao osnovu koristio protokol TypeScript jezičkog servera, koji je uopšten i napravljen da bude jezički neutralan. U početku, protokol je podržavao poruke dijagnostike, a kasnije je obogaćen ostalim "pametnim" funkcionalnostima koje jedan server može da pruži.

Danas je LSP podržan od strane velikog broja tekst editora i jezičkih zajednica. LSP je otvoren protokol i kompletna specifikacija je slobodno dostupna na GitHub-u [<https://github.com/Microsoft/language-server-protocol>].

Razvojem standardizovanog protokola, postalo je dovoljno da jezička zajednica napravi jedan jezički server koji pruža sve "pametne" funkcionalnosti za dati programski jezik. Svi editori koji razumeju jezički protokol tada automatski dobijaju podršku za taj programski jezik.

1.2 Tehnički detalji

Tekst editor se ponaša kao klijent koji komunicira sa jezičkim serverom. Komunikacija se odvija slanjem poruka.

U osnovi, poruka se sastoji iz **zaglavlja** (*header*) i **sadržaja** (*content*), slično kao i HTTP protokol. Zaglavlje i sadržaj su razdvojeni `\r\n` karakterima.

Specifikacijom nije određen transportni protokol za razmenu poruka.

U praksi klijent i server najčešće komuniciraju preko standardnog ulaza/izlaza, ali moguća je i komunikacija preko mrežnih soketa ili bilo kojeg drugog vida međuprocene komunikacije.

1.2.1. Zaglavlje poruke

Zaglavlje se sastoji iz polja. Svako polje je par ključ-vrednost, razdvojeni sa : (dvotačka praćena razmakom). Svako polje se završava karakterima `\r\n`. Podržana polja u zaglavlju su `Content-Length` i `Content-Type`.

1.2.2. Sadržaj poruke

Sadržaj poruke koji se razmenjuje između klijenta i servera je baziran na JSON-RPC protokolu. JSON-RPC je protokol koji omogućava udaljen poziv procedura (*remote procedure call*) razmenom poruka u JSON formatu.

Sve poruke se dele na tri varijante:

1. **Zahtev** (*request*)—upućen zahtev između klijenta i servera. Svaki zahtev mora imati odgovor.
2. **Odgovor** (*response*)—odgovor na upućen zahtev.
3. **Obaveštenje** (*notification*)—slično zahtevu, ali ne zahteva odgovor.

1.2.3. Poruke za sinhronizaciju teksta

Tekst editor obaveštava jezički server o događajima, odnosno interakcijama koje korisnik vrši nad dokumentima.

Neki od mogućih događaja su:

- `didOpen`—Korisnik je otvorio datoteku.
- `didChange`—Korisnik je napravio izmene u datoteci.
- `didSave`—Korisnik je sačuvao datoteku.
- `didClose`—Korisnik je zatvorio datoteku.

Jezički server treba da "sluša" događaje i interno vodi evidenciju o svim otvorenim datotekama i njihovim sadržajem.

Kada se dogodi određeni događaj, klijent u poruci ne šalje samo URI dokumenta već njegov kompletan sadržaj u vidu stringa. Ovo je neophodno kako bi sve funkcionalnosti radile bez potrebe da korisnik prethodno sačuva datoteku na medijum za trajno čuvanje podataka.

1.2.4. Poruke za objavljivanje dijagnostike

Poruke dijagnostike imaju za cilj da korisniku prikažu greške, upozorenja ili informacije u toku samog unosa teksta. Poruke dijagnostike ne traži klijent od servera, već ih server sam isporučuje kada je to neophodno. Iz tog razloga ovakve poruke su implementirane kao obaveštenja, a ne kao par zahtev–odgovor.

2. PROGRAMSKI JEZIK MINIC

Programski jezik miniC je nastao na Fakultetu tehničkih nauka, Univerziteta u Novom Sadu, za potrebe kursa Programski prevodioci.

miniC je striktan podskup programskog jezika C. Dakle, miniC programe je moguće kompajlirati regularnim C kompajlerom, dok obrnuto ne važi. Nastao je odabirom osobina i koncepata C jezika koji su interesantni za implementaciju kompajlera. Međutim, ove osobine su uzete u određenoj meri kako bi se olakšala implementacija kompajlera. Autori su se uzdržali od mnogih karakteristika C jezika koji nepotrebno komplikuju implementaciju kompajlera, a koji, u edukativnom smislu, ne doprinose značajno.

3. OPIS SOFTVERSKOG REŠENJA

U ovom poglavlju je opisan način implementacije jezičkog servera za programski jezik miniC. Takođe, dat je opis alata i biblioteka koje su korišćene prilikom razvoja opisanog softverskog rešenja.

3.1 Opis korišćenih alata i biblioteka

3.1.1. Flex

Flex [6] [7] je program koji generiše leksički analizator, odnosno skener. Flex čita ulazna pravila i kao izlaz vraća kod koji implementira leksar u programskom jeziku C. Nastao je 1987. godine kao slobodna verzija programa Lex. Dostupan je pod modifikovanom BSD licencom, a njegov originalni autor je Vern Paxson.

Zadatak leksičkog analizatora je skeniranje sekvence tokena u ulaznom tekstu i pronalaženje određenih šablona. Kada se šablon prepozna, izvršava se zadata akcija. Šabloni se zadaju u vidu regularnih izraza. Flex koristi sebi specifičnu sintaksu za zadavanje regularnih izraza, ali je ona u velikoj meri slična uobičajenoj ERE sintaksi koja je definisana u IEEE POSIX standardu.

3.1.2. Bison

Bison [6] [8] je program koji generiše sintakсни analizator, odnosno parser. Bison čita ulazna pravila i kao izlaz vraća kod koji implementira parser u programskom jeziku C. Nastao je 1985. godine kao slobodna verzija programa Yacc. Dostupan je kao deo GNU projekta, pod GNU opštom javnom licencom, verzija 3.

Sintaksnom analizom se odgovara na pitanje "Da li je raspored tokena u skladu sa gramatikom?". Gramatika se zadaje u BNF notaciji (Bakus-Naurova forma).

Bison se često koristi u kombinaciji sa Flex-om za izradu kompajlera. Leksičkom analizom se prepoznaju šabloni u tekstu i tekst se deli na tokene, a zatim se sintaksnom analizom proverava da li se tokeni nalaze u ispravnom redosledu.

3.1.3. cJSON biblioteka za parsiranje JSON-a

cJSON [9] je biblioteka napisana u programskom jeziku C namenjena parsiranju JSON-a. S obzirom da LSP definiše razmenu poruka između klijenta i servera preko JSON

objekata, cJSON je ključna biblioteka za implementaciju jezičkog servera opisanog u nastavku. cJSON je nastao 2009. godine, a originalni autor biblioteke je Dave Gamble. Biblioteka je dostupna pod MIT Expat licencom.

3.2. Opis implementacije jezičkog servera minic-lsp za programski jezik miniC

S obzirom da jezički server mora poznavati gramatiku jezika kao i sam kompajler, za implementaciju minic-lsp jezičkog servera uzeta je kao osnova postojeći kompajler miniC jezika pod imenom MICKO. Kompajler MICKO je nastao na Fakultetu tehničkih nauka za potrebe edukacije na predmetu Programski prevodioci.

Prvu dodatnu stvar koju treba implementirati kako bi jezički server funkcionisao je sama obrada poruka i slanje odgovora. Pošto sam LSP ne definiše transportni protokol za razmenu poruka, potrebno je odlučiti koji pristup će se koristiti. Zbog jednostavnosti implementacije izabrana je komunikacija preko standardnog ulaza/izlaza.

Po startovanju programa, poziva se `lsp_event_loop` funkcija koja startuje beskonačnu petlju za obrađivanje poruka.

```
for(;;) {
    unsigned long content_length =
        lsp_parse_header();
    cJSON *request =
        lsp_parse_content(content_length);
    json_rpc(request);
    cJSON_Delete(request);
}
```

Funkcija `lsp_parse_header` preuzima zaglavlje poruke i parsira dužinu sadržaja koji sleduje. Funkcija `lsp_parse_content` preuzima sadržaj specificirane dužine `content_length` i kao rezultat vraća parsiran JSON zahtev/obaveštenje. Funkcija `json_rpc` parsira udaljeni poziv procedure po JSON-RPC protokolu i poziva odgovarajuću funkciju. Na kraju je potrebno osloboditi zauzetu memoriju pozivom `cJSON_Delete` funkcije.

4. INTEGRACIJA SERVERA SA TEKST EDITORIMA

Samim tim što je implementiran jezički server za miniC programski jezik, svi editori teksta koji poznaju zajednički LSP jezik automatski dobijaju podršku za miniC.

Da bi korisnik počeo da koristi jezički server, potrebno je da uradi sledeće:

1. Jezički server treba instalirati na neku od putanja u `$PATH` varijabli.
2. U podešavanjima tekst editora treba specificirati koji jezički serveri se koriste za koje programske jezike. Editoru je potreban samo naziv izvršne datoteke jezičkog servera, kako bi znao koji program da startuje. Nikakve dodatne informacije editoru nisu potrebne, jer svi serveri pričaju zajednički LSP jezik.

Da korisnik ne bi morao ručno da podešava tekst editor da koristi minic-lsp, u sklopu ovog rada napravljena je konfiguracija u vidu plugin-a za tekst editore Vim i Atom. Vim [10] je konfigurabilni tekst editor namenjen efikasnom kreiranju i modifikovanju tekst dokumenata. Nastao je 1991. godine, a originalni autor je Bram Moolenaar. Vim je slobodni softver dostupan pod licencom koja ohrabruje korisnike da doniraju novac deci u Ugandi.

U vreme pisanja ovog rada, poslednja dostupna verzija Vim-a je 8.1 i ne postoji ugrađena podrška za LSP. Ipak, pošto je Vim proširiv kroz plugin-e, postoje razna proširenja pomoću kojih Vim postaje LSP klijent. Plugin koji je izabran za ovu demonstraciju se zove ALE (Asynchronous Lint Engine) [11]. ALE je takođe slobodni softver dostupan pod BSD licencom sa dve klauzule.

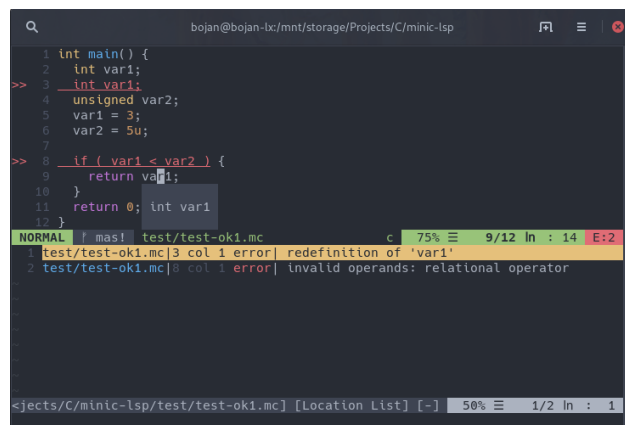
Atom [12] je "hakabilni tekst editor 21. veka" napravljen od strane Github kompanije 2014. godine. Atom je slobodni softver dostupan pod MIT Expat licencom. Baziran je na Electron platformi za razvijanje aplikacija.

U vreme pisanja ovog rada, poslednja dostupna verzija Atom-a je 1.40 i ne postoji ugrađena podrška za LSP. Međutim, Atom je takođe proširiv kroz plugin-e i dostupno je proširenje pomoću kojeg Atom postaje LSP klijent. Plugin koji Atom pretvara u LSP klijent se zove `atom-ide` [13] i razvijen je od strane Facebook kompanije. Dostupan je kao slobodni softver pod BSD licencom sa tri klauzule.

5. DEMONSTRACIJA RADA JEZIČKOG SERVERA MINIC-LSP

Slika 1 prikazuje Vim tekst editor sa otvorenom test datotekom. Linije 3 i 8 su podvučene crvenom linijom, a sa leve strane se vide znakovi `>>`, što označava grešku u navedenim linijama. U donjoj polovini editora vidljiva je lista svih dijagnostika za otvorenu datoteku.

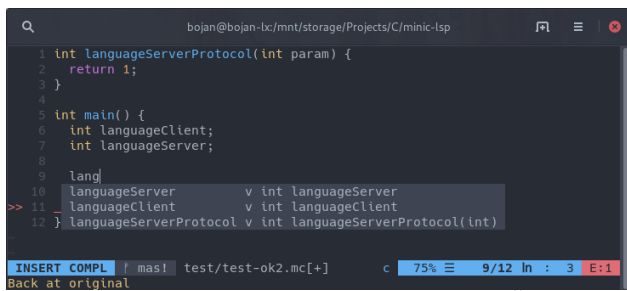
Takođe, može se videti da je korisnik postavio kursor na liniju 9, na slovo `r` u identifikatoru `var1`, a zatim je zatražio informacije u vidu "lebdeće" poruke. Na osnovu informacija koje su vidljive, korisnik može saznati da je `var1` tipa `int`. U slučaju da identifikator na kojem se nalazi kursor predstavlja naziv funkcije, vidljivi su tipovi parametara koje funkcija prihvata kao i tip povratne vrednosti funkcije.



SLIKA 1. DEMONSTRACIJA RADA DIJAGNOSTIKA

Slika 2 prikazuje Vim tekst editor sa otvorenom test datotekom koja sadrži dve funkcije. U funkciji `main` su definisane dve lokalne promenljive. Korisnik u liniji 6 unosi karaktere `lang`, nakon čega tekst editor automatski otvara padajući meni sa dostupnim opcijama koje se mogu dopuniti.

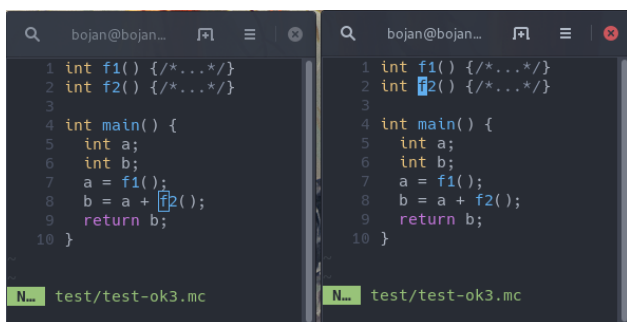
U padajućem meniju se pored teksta dopune vidi i kako izgleda deklaracija simbola. Na osnovu deklaracije korisnik može saznati kojeg tipa su promenljive, koje parametre prihvata funkcija, kojeg tipa je povratna vrednost funkcije...



```
1 int languageServerProtocol(int param) {
2     return 1;
3 }
4
5 int main() {
6     int languageClient;
7     int languageServer;
8
9     lang
10    languageServer v int languageServer
11    languageClient v int languageClient
12 } languageServerProtocol v int languageServerProtocol(int)
```

SLIKA 2. DEMONSTRACIJA RADA DOVRŠAVANJA

Slika 3 prikazuje funkcionalnost skoka na definiciju. Sa leve strane je prikazana inicijalna pozicija kursora. Korisnik je pozicionirao kursor na poziv `f2` funkcije. Korisnik inicira komandu skoka na definiciju i kursor se prebacuje na poziciju gde je funkcija definisana, kao što se može videti na desnoj polovini slike.



```
1 int f1() { /*...*/ }
2 int f2() { /*...*/ }
3
4 int main() {
5     int a;
6     int b;
7     a = f1();
8     b = a + f2();
9     return b;
10 }
```

SLIKA 3. DEMONSTRACIJA RADA SKOKA NA

6. ZAKLJUČAK

Korisnici danas imaju visoka očekivanja od tekst editora. Iz tog razloga, tekst editori su veoma kompleksni programi, sa velikim brojem mogućnosti. Pritom, većina funkcionalnosti je usko povezana sa programskim jezikom u kojem je dokument napisan.

Kako programskih jezika vremenom ima sve više, tako je i razvoj tekst editora postao sve kompleksniji. Rešenje ovog problema je razvoj posebnih jezičkih servera koji pružaju "pametne" funkcionalnosti za pojedinačne programske jezike. Tekst editor više ne mora da poznaje sintaksu ili semantiku jezika, već samo treba da se integriše sa jezičkim serverom.

Međutim, razdvajanjem tekst editora od jezičkih servera, rešen je jedan problem, ali je nastao novi problem—njihova integracija. Jezički serveri su koristili različite formate poruka za komunikaciju. Tekst editori su morali na različite načine parsirati poruke koje dobijaju od jezičkih servera, a morali su poznavati i različite poruke za zadavanje upita. Kako bi se problem integracije rešio, Microsoft je razvio protokol jezičkih servera. Razvojem zajedničkog jezika kojim jezički serveri komuniciraju, postalo je moguće da tekst editori napišu integracioni kod samo jednom i da ga koriste za sve postojeće i buduće programske jezike.

Razvojem LSP-a, razlika između običnih tekst editora i integrisanih razvojnih okruženja postaje sve manja. Tekst editori podržavaju većinu funkcionalnosti i jezički su

neutralni, pa nestaje potreba za specijalizovanim integrisanim razvojnim okruženjima. Korisnici mogu da koriste tekst editor sa kojim su već upoznati za sve projekte, bez obzira na programski jezik koji se koristi.

Pisanje kompletnog softverskog rešenja u okviru ovog rada je olakšano uz pomoć LSP-a. Korišćen je Vim tekst editor i Cland jezički server za programski jezik C.

6. LITERATURA

- [1] Official page for Language Server Protocol, <https://microsoft.github.io/language-server-protocol/>, Avgust 2019.
- [2] A community-driven source of knowledge for Language Server Protocol implementations, <https://langserver.org/>, Avgust 2019.
- [3] Language server protocol history, <https://github.com/Microsoft/language-server-protocol/wiki/Protocol-History>, Avgust 2019.
- [4] Zorica Suvajdzin Rakić i Miroslav Hajduković, miniC, specifikacija i implementacija, zbirka zadataka, Univerzitet u Novom Sadu, FTN, Edicija tehničke nauke - udžbenici, broj ???, FTN Izdavaštvo, Novi Sad, 2014
- [5] Zorica Suvajdzin Rakić, Predrag Rakić, Tara Petrić, miniC Project for Teaching Compilers Course, ICIST 2014
- [6] Zorica Suvajdzin Rakić i Predrag Rakić, flex & bison, zbirka zadataka, Univerzitet u Novom Sadu, FTN, Edicija tehničke nauke - udžbenici, broj 478, FTN Izdavaštvo, Novi Sad, 2014
- [7] Flex zvanična dokumentacija, <https://westes.github.io/flex/manual/>, Avgust 2019.
- [8] Bison zvanična dokumentacija, <https://www.gnu.org/software/bison/manual/>, Avgust 2019.
- [9] cJSON repozitorijum, <https://github.com/DaveGamble/cJSON>, Avgust 2019.
- [10] Vim tekst editor, <https://www.vim.org/>, Avgust 2019.
- [11] ALE plugin za Vim tekst editor, <https://github.com/dense-analysis/ale>, Avgust 2019.
- [12] Atom tekst editor, <https://atom.io/>, Avgust 2019.
- [13] Atom-IDE plugin za Atom tekst editor, <https://ide.atom.io/>, Avgust 2019.

Kratka biografija:

Bojan Stipić rođen je u Novom Sadu 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Računarstvo i informatika odbranio je 2019.god. Kontakt: bojanstipic@uns.ac.rs

АУТОМАТСКО ТЕСТИРАЊЕ МИКРО СЕРВИСА АЛАТИМА ЗА ТЕСТИРАЊЕ ПРИХВАТЉИВОСТИ**AUTOMATIC TESTING OF MICRO SERVICES WITH TOOLS FOR ACCEPTANCE TESTING**

Никола Ђуза, Факултет техничких наука, Нови Сад

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – У овом раду је извршена анализа и поређење тренутно најпопуларнијих JavaScript алата за тестирање прихватљивости веб страница. За анализу алата је коришћен пројекат за продају међународног путног осигурања који се аутоматски тестира са свим алатима који су анализирани. Пројекат који се тестира поседује микро сервисну архитектуру, где се сваки сервис налази унутар свог контејнера. Рад улази у детаље писања и покретања аутоматизованих тестова прихватљивости уз алате, како у локалном окружењу, тако и у окружењима за континуалну интеграцију.

Кључне речи: Аутоматско тестирање, тестирање прихватљивости, микро сервисна архитектура

Abstract – This thesis analyzes and compares most popular JavaScript tools for acceptance testing. A project for selling travel insurance with automated tests has been used to analyze all the tools. Project that's being tested is micro service based and every service is inside its own container. Thesis goes into details on how to write and run tests using these tools, both in local environment and in continuous integration environments.

Keywords: Automated testing, acceptance testing, micro service architecture

1. УВОД

Тестирање је важна активност у развоју софтвера која чини развој софтвера више поузданим. Тестирање добија на значају све више како расте вредност софтвера. Избегавање тестирања софтвера често доводи до грешака у софтверу које могу проузроковати велике новчане штете. Из овог разлога се у последње време не штеди на тестирању и активностима које оно подразумева.

Како комплексност и вредност софтвера расте, тако расту и начини на које се тај софтвер може тестирати. Поред мануелног тестирања, доста је заступљено аутоматско тестирање. Аутоматско тестирање помаже у одржавању поузданости и једноставности програмског кода јер се може покретати приликом сваке измене у изворном коду софтвера у процесу итеративног развоја софтвера.

НАПОМЕНА:

Овај рад је произтекао из мастер рада чији ментор је био др Бранко Милосављевић, ред. проф.

Аутоматски тестови често помажу у проналажењу грешака пре него што се то деси од стране корисника софтвера, и то у раним фазама развоја. Једна врста аутоматских тестова су интеграциони тестови који тестирају софтвер на сличан начин на који корисник треба да користи софтвер. Овај рад ће објаснити које активности и начини се могу користити како би се тестирао сет микро сервиса уз помоћ алата за тестирање прихватљивости. Такође ће фокус бити на аутоматским тестовима, како их писати и покретати, како у локалном, тако и у другим окружењима.

1.1 Шта је тестирање?

Тестирање се дефинише као активност којом се проверава да ли добијени резултат одговара очекиваном резултату и да ли је систем без грешака. Тестирањем се такође врши верификација софтвера и да ли он извршава оно за шта је дефинисан. Битно је напоменути да је тестирањем готово немогуће открити све грешке у систему [1]. Разлог за ово је јер не можемо предвидети све гране извршавања софтвера или постоји пропуст у дизајну софтвера. За разлику од физичких система, већина дефеката у софтверу се дешава због грешке у дизајну, а не у процесу израде.

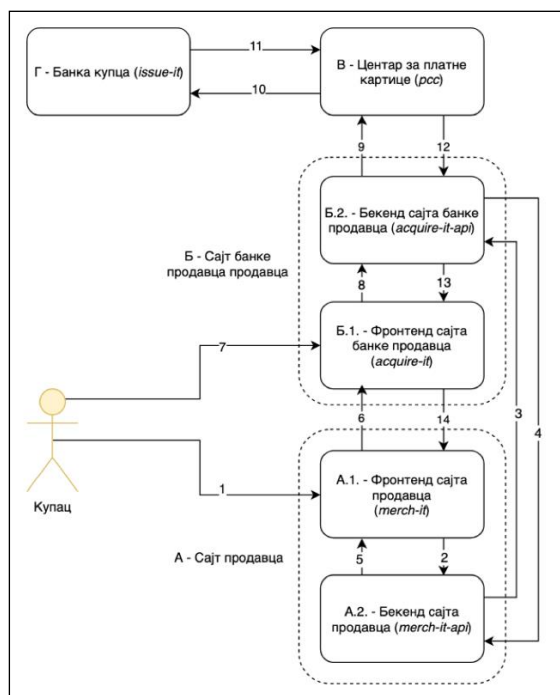
2. СПЕЦИФИКАЦИЈА ПРОЈЕКТА

У овом поглављу ће се описати пројекат над којим су тестирани алати за тестирање прихватљивости. У питању је пројекат са сложенем архитектуром микро сервиса. Пројекат је рађен у Ruby и JavaScript програмским језицима. Такође је коришћена Docker алат за креирање софтверских контејнера. За имплементацију корисничког интерфејса је коришћен JavaScript језик и React библиотека. За израду бекенд дела је коришћен Ruby програмски језик и Ruby on Rails фрејмворк. Као база података за чување података је коришћена PostgreSQL систем за управљање базама података.

2.1 Архитектура пројекта

Пројекат је сервис за продају путног осигурања корисницима. Сценарио који се разматра у слици 1. је следећи: Купац који има отворени рачун у банци Б (Банка купца) врши неку куповину међународног путног осигурања преко корисничког интерфејса код Продавца и плаћа платном картицом посредством веб апликације Банке продавца за наплату на интернету. Рачун продавца се налази у банци А (Банка продавца).

Комуникација између банке купца (банка Б) и банке продавца (банка А) врши се посредством Центра за платне картице. На слици 1. су приказани сви сервиси који постоје у овом пројекту, као и комуникација описана у претходном примеру куповине међународног осигурања.



Слика 1. Дијаграм тока плаћања са рачуна из банке купца

2.2 Контејнеризација уз помоћ Docker алата

У традиционалном развоју софтвера, програмски код и софтвер који је развијен у једном окружењу често зна да има доста дефеката кад се покрене на неком другом систему. Инжењери софтвера ово решавају тако што се софтвер покреће у контејнерима у облаку. Контејнери пружају логичко паковање у ком је софтвер на неки начин издвојен од окружења у ком заправо ради. Овакво раздвајање дозвољава апликацијама које су у контејнерима да се лакше лансирају на друго платформи, небитно да ли је у питању приватни дата центар, јавни облак, или чак на рачунару свог колеге. Идеја је да имплементација апликација не зависи од окружења у ком се налази.

Контејнери се често пореде са виртуалним машинама (енгл. *virtual machine* – VM). Виртуалне машине су гостујући оперативни систему као што је *Linux* или *Windows* који је покренут на домаћинском оперативном систему са контролисаним приступом хардверу домаћина. Слично виртуалним машинама, контејнери дозвољавају да се апликације запакују са свим библиотекама и осталим потребним софтвером, пружајући изоловано окружење за покретање саме апликације (сервиса).

3. АНАЛИЗА АЛАТА ЗА ТЕСТИРАЊЕ ПРИХВАТЉИВОСТИ

У овом поглављу ће бити коришћен претходно описан пројекат како би се анализирали најпопуларнији алати (често се називају и фрејмворци и библиотеке) за

тестирање прихватљивости унутар *JavaScript* заједнице. Алати који се користе најчешће данас су *Cypress*, *Puppeteer*, *Nightwatch.js*, *TestCafe*, *CodeceptJS* [2] и њих ћемо анализирати и касније упоредити. Биће искоришћен претходно описани пројекат за продају осигурања како би се написали и покретали тестови прихватљивости тј. *e2e* тестови. Тестови ће бити покретани у локалном окружењу, као и у окружењу на два позната сервиса за континуалну интеграцију – *Semaphore CI* и *Travis CI*.

3.1 Cypress

Cypress је *JavaScript* фрејмворк за *e2e* тестирање. Тренутно је најпопуларнија библиотека за *e2e* тестирање [3]. Главна разлика у односу на конкуренцију је што не користи *Selenium* за тестирање. Већина библиотека користи *Selenium* и то је узрок њихових мана у већини случајева. Како *Selenium* покреће команде преко мреже и контактира претраживач споља, *Cypress* се налази унутар претраживача и одатле покреће тестове.

Главни фокус *Cypress* алата су *e2e* тестови и сав рад је усмерен на побољшавање *e2e* тестова. Да би ово омогућили, *Cypress* је направљен тако да може да се користи са било којом *JavaScript* библиотеком или фрејмворком за развој графичком интерфејса. Сви тестови који се пишу унутар *Cypress* фрејмворка морају да се пишу у *JavaScript* програмском језику. Разлог за ово је што се *Cypress* покреће и ради унутар претраживача, а претраживач најбоље извршава *JavaScript* код.

Велика предност је што кад се *Cypress* инсталира у оквиру пројекта као библиотека, долази упакован са свих потребних алатима за *e2e* тестирање. Ово драстично смањује време потребно за конфигурацију и инсталацију алата који већ долазе упаковани и спремни за рад од почетка. На слици 2. се налази пример теста написаног у *Cypress* фрејмворку.

```

1 describe('Paying for plans', () => {
2   it('cheapest one', () => {
3     cy.visit('http://localhost:3000')
4
5     cy.contains('Welcome to merch-it')
6
7     cy.get('[data-cy=email]').type('test@example.com')
8     cy.get('[data-cy=10000]').click()
9
10    cy.contains('Checkout')
11    cy.get('[data-cy=submit]').click()
12
13    cy.get('[data-cy=pan]').type('1')
14    cy.get('[data-cy=securityCode]').type('1')
15    cy.get('[data-cy=cardHolderName]').type('1')
16    cy.get('[data-cy=expirationDate]').type('1')
17
18    cy.get('[data-cy=pay]').click()
19
20    cy.contains('Congrats, you've bought insurance from us!')
21  })
22 })

```

Слика 2. Пример *Cypress* тест фајла

3.2 Puppeteer

Puppeteer је *Node.js* библиотека која пружа API високог нивоа за контролисање *Chrome* и *Chromium* преко *Chrome DevTools* протокола. Не плаћа се и *open source* је библиотека доступна на сајту *GitHub*.

Puppeteer се покреће без графичког интерфејса (енгл. *headless*) али се може подесити да ради са графичким интерфејсом у *Chrome* и *Chromium* претраживачима. Он је решење које користи сам *Chrome* претраживач при тестирању њихових функционалности.

Chrome DevTools протокол је сет алата за мерење, инспекцију, проналажење грешака у коду (енгл. *debugging*) и профилисање *Chromium* и *Chrome*. Пример теста написан у *Puppeteer* библиотеци је дат на слици 3.

```
1 test("Balance is too low", async () => {
2   const browser = await puppeteer.launch();
3   const page = await browser.newPage();
4   await page.goto("http://localhost:3000");
5
6   await page.type("[data-cy=email]", "test@example.com");
7   await page.click("[data-cy='10000']");
8   await page.click("[data-cy=submit]");
9
10  await page.waitForSelector("[data-cy=pan]");
11  await expect(page).toMatch("PAN");
12
13  await page.type('[data-cy=pan]', '4');
14  await page.type('[data-cy=securityCode]', '4');
15  await page.type('[data-cy=cardHolderName]', '4');
16  await page.type('[data-cy=expirationDate]', '4');
17
18  await page.click('[data-cy=pay]');
19
20  await page.waitForSelector("[data-cy=errorBox]");
21  await expect(page)
22    .toMatch("Oops! The transaction didn't go through, sorry.");
23
24  await browser.close();
25 });
```

Слика 3. Пример *Puppeteer* тест фајла

3.3 Nightwatch.js

Nightwatch.js је решење за *e2e* тестирање базирано на *Node.js* библиотеци за претраживаче и веб сајтове. Бесплатан је и *open source*. За комуникацију са претраживачем користи моћан *W3C WebDriver API* како би извршавао команде и провере над елементима у *DOM* стаблу веб страница. *W3C WebDriver API* или скраћено *WebDriver* је интерфејс за даљинску контролу претраживача. Он пружа протокол који не зависи од платформе или програмског, већ дозвољава да се даљински диктира понашање веб претраживача [21]. Пример теста написан у *Nightwatch.js* библиотеци је дат на слици 4.

```
1 module.exports = {
2   tags: ['insurance'],
3
4   'Pay for first option': function(browser) {
5     browser
6       .url('http://localhost:3000')
7       .waitForElementVisible('body', 1000);
8
9     browser.setValue("[data-cy=email]", "test@example.com")
10      .click("[data-cy='10000']")
11      .click("[data-cy=submit]")
12      .waitForElementVisible("[data-cy=pan]")
13      .setValue("[data-cy=pan]", '2')
14      .setValue("[data-cy=securityCode]", '2')
15      .setValue("[data-cy=cardHolderName]", '2')
16      .setValue("[data-cy=expirationDate]", '2')
17      .click("[data-cy=pay]")
18      .assert.containsText("[data-cy=successBox]",
19        "Congrats, you've bought insurance from us!");
20
21     browser.end();
22   }
23 };
```

Слика 4. Пример *Nightwatch.js* тест фајла

3.4 TestCafe

TestCafe је бесплатан и *open source Node.js* алат који служи за аутоматизовање *e2e* веб тестова. Тестови се могу писати у *JavaScript* или *TypeScript* програмским језицима. *TestCafe* ради на свим популарним оперативним системима као што су *Windows*, *Mac OS*, и *Linux*. Такође постоји подршка за тестирање мобилних, десктоп, удаљених (енгл. *remote*) и претраживача у облаку (енгл. *cloud*).

За покретање тестова није потребан ни *WebDriver* ни *Selenium*. *TestCafe* користи други приступ при контролисању претраживача, а то је *URL-rewriting* проху механизам. Користећи овај механизам, акције корисника у тесту су емулиране кроз *DOM API* који је доступан преко драјвер скрипте која је додата путем *proxy*-ја на страницу. На овај начин, подешавања окружења за покретање *TestCafe* тестова је прилично једноставно. Све што је потребно је да се инсталира *TestCafe* пакет са *npm* регистра *JavaScript* пакета у пројекат. На слици 5 је приказан пример теста написан у *TestCafe* алату.

```
1 import { Selector } from 'testcafe';
2
3 fixture 'Insurance'
4   .page 'http://localhost:3000';
5
6 test('Balance is too low', async t => {
7   await t
8     .typeText('[data-cy=email]', 'test@example.com')
9     .click("[data-cy='10000']")
10    .click('[data-cy=submit]')
11
12    .typeText('[data-cy=pan]', '4')
13    .typeText('[data-cy=securityCode]', '4')
14    .typeText('[data-cy=cardHolderName]', '4')
15    .typeText('[data-cy=expirationDate]', '4')
16
17    .click('[data-cy=pay]')
18    .expect(Selector('[data-cy=errorBox]').innerText).contains(
19      "Oops! The transaction didn't go through, sorry."
20    );
21 });
```

Слика 5. Пример *TestCafe* тест фајла

3.5 CodeceptJS

CodeceptJS је фрејмворк за тестирање који користи друге популарне библиотеке за тестирање и додаје *API* високог нивоа који се користи за писање тестова и који повећава читљивост *e2e* тестова. Библиотеке које су укључене су:

- *Protractor* – користи се ова библиотека путем *WebDriver* протокола
- *Puppeteer* – користи се брзи *headless* режим са *Google Chrome* претраживачом за покретање тестова
- *Appium* – библиотека која се користи за тестирање мобилних уређаја
- *Nightmare* – користи *Electron* и *NightmareJS* за покретање тестова

CodeceptJS такође нуди опцију покретања тестова користећи *WebDriver* протокол кроз *webdriverio* библиотеку. Ово је фрејмворк који изолује и прави апстракт од набројаних библиотека, дозвољавајући кориснику да пише тестове у разумљивом језику. Ове библиотеке се зову помагачи (енгл. *helpers*). На слици 6 је приказан пример теста у *CodeceptJS* фрејмворку.

```

1 Feature('Pay for insurance');
2
3 Scenario('Pay for first option', (I) => {
4   I.amOnPage('http://localhost:3000')
5   I.fillField('[data-cy=email]', 'test@example.com')
6   I.click('[data-cy=10000]')
7   I.click('[data-cy=submit]')
8
9   I.waitForText('PAN');
10
11  I.fillField('[data-cy=pan]', '2')
12  I.fillField('[data-cy=securityCode]', '2')
13  I.fillField('[data-cy=cardHolderName]', '2')
14  I.fillField('[data-cy=expirationDate]', '2')
15  I.click('[data-cy=pay]')
16
17  I.waitForText("Congrats, you've bought insurance from us!");
18 });

```

Слика 6. Пример CodeceptJS тест фајла

4. РЕЗУЛТАТИ АНАЛИЗЕ

На слици 7. су упоређени алати и четири категорије које их карактеришу. Прва категорија је лакоћа подешавања где је узето у обзир колико је лако инсталирати алат у локалном окружењу, али и на сервисима за континуалну интеграцију. Друга категорија је лакоћа писања тестова где се посматра колико је лако савладати синтаксу која долази уз алат, колико је она документована и општи утисак о писању тестова. Ове две категорије носи максимално 5 поена. У трећој категорији се гледа да ли алат подражава тестирање на више претраживача и носи 1 поен, а четврта категорија прати да ли се тестови могу паралелно покретати уз алат и такође носе 1 поен.

	Лакоћа подешавања	Лакоћа писања тестова	Подршка за више претраживача	Паралелно покретање тестова
 Cypress	5/5	5/5		
 Puppeteer	4/5	3/5		
 Nightwatch.js	4/5	3/5		
 TestCafe	5/5	4/5		
 CodeceptJS	3/5	5/5		

Слика 7. Табела поређења анализираних алата за тестирање прихватљивости

На основу табеле, може се закључити да на основу оцењених категорија *TestCafe* и *Cypress* односе победу што се тиче најбољих алата за *e2e* тестирање и тестирање прихватљивости. У даљем тексту ће бити објашњен резултат поређења сваког од алата у табели.

Cypress има одличне оцене за подешавање и писање тестова, али нажалост не подржава тестирање са више претраживача (*cross-browser testing*). Ова функционалност је најављена да ће бити имплементирана у будућности. Одлично се показао приликом подешавања на сервисима за континуалну интеграцију. Алат који олакшава *TDD* приступ развоју софтвера јер

аутоматски покреће тестове при променама у тестовима пројекта. Одличан за развој иначе јер се покреће у претраживачу и може бити покренут док се развија функционалност и одмах се проверити да ли софтвер ради или не.

Puppeteer на овој табели има најлошију позицију, али то не значи да не постоји случај у ком би се овај алат могао користити. *Puppeteer* специјализује у покретању тестова у *Google Chrome* претраживачу, и у пројекту где је тај претраживач једини захтев, заједно са брзином покретања тестова, *Puppeteer* би могао бити прави избор. Такође његова могућност да се конфигурише на различите начине и слобода која је тиме дата могу се показати као добар избор за одређене тимове који већ имају подешен систем на одређен начин.

Nightwatch.js је добар избор ако је потребно тестирати на више претраживача користећи *WebDriver* протокол јер се *Selenium* не препоручује у тренутној верзији алата. У овој табели, на основу резултата поређења, *Nightwatch.js* заузима претпоследње место.

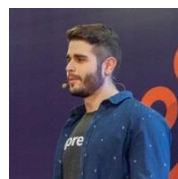
TestCafe се показао као најкомплетније решење које тренутно постоји у свету *JavaScript e2e* тестирања јер је лак за подешавање и писање тестова, подржава тестирање са више претраживача, и све то може да се извршава у паралели. Одличан избор ако је у пројекту потребно да се тестира софтвер на више претраживача јер његов директни ривал – *Cypress* не подржава ту функционалност.

CodeceptJS је одличан алат који пружа читљиву синтаксу за писање тестова. Подржава више претраживача али је тежи да се подеси у односу на остале алате. Такође постоје нерешене разлике у командама које корисник може да зада у тестовима у зависности од претраживача за који се определи. Због овога *CodeceptJS* заузима друго место на основу резултат поређења у табели. Одличан је избор ако је тимовима потребно да дефинишу читљиве тестове за остатак тима, али из разлога што користи друге алате као помагаче, није успео да се више пласира.

5. LITERATURA

- [1] J. Pan, Software Testing (coursework), Carnegie Mellon University, 1999.
- [2] V. Zaidman, "An Overview of JavaScript Testing in 2019," WellDone Software, 18 February 2019.
<https://medium.com/welldone-software/an-overview-of-javascript-testing-in-2019-264e19514d0a>
- [3] „The State of JavaScript 2018: Testing – Other Libraries,”
<https://2018.stateofjs.com/testing/other-libraries/>

Кратка биографија:



Никола Буза рођен је у Новом Саду 1993. године. Мастер студије на Факултету техничких наука из области Електротехнике и рачунарства – примењене рачунарске науке уписао је 2016. године након завршених основних академских студија из исте области. контакт: nikoladjuza@uns.ac.rs

PC APLIKACIJA ZA EDUKATIVNI ELEKTROFIZIOLOŠKI SISTEM**A PC APPLICATION FOR AN EDUCATIONAL ELECTROPHYSIOLOGICAL SYSTEM**

Aleksa Mitreski, *Fakultet tehničkih nauka, Novi Sad*

Oblast: RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – Ovaj rad daje uvid u jedno softversko rešenje za elektrofiziološki sistem koji bi se koristio u edukativne svrhe za studente biomedicinskog inženjerstva. Cilj rada je da se preko elektrofiziološkog akvizicionog sistema prikupe bioelektrični signali u vidu digitalnih podataka koji bi se nadalje slali računarskoj aplikaciji radi prikaza, čuvanja i obrade istih. Aplikacija je na kraju bila testirana.

Ključne reči: PC aplikacija, elektrofiziološki sistem, edukativni sistem

Abstract – The paper offers an insight to a new software solution for an electrophysiological system which would be used in educational purposes. The goal of the project was to use an electrophysiological data acquisition system which would transfer data to the PC application so that it could be viewed, stored and processed. In the end, the application was evaluated.

Keywords: PC application, electrophysiological system, educational system

1. UVOD

Tradicionalne metode prenošenja znanja i veština u savremenom dinamičnom dobu gube smisao, te bivaju zamenjene novim edukativnim tehnologijama, koje se mogu definisati na sledeći način :

„Edukativne tehnologije predstavljaju izučavanje i primenu poboljšanog podučavanja stvaranjem, upravljanjem i upotrebom adekvatnih tehničkih sredstava i procesa [1].“ Kao i svi aspekti tehnologije i edukativna je doživela nagli razvoj poslednjih decenija i sve se više primenjuje u obrazovnim sistemima širom sveta.

Međutim, kako u nauci, tako i u tehnologiji, nastaju nove istraživačke grane, u kojima se otvaraju nove mogućnosti zbog kojih i edukativne tehnologije doživljavaju još intenzivniji razvoj. Jedna takva grana je biomedicinsko inženjerstvo. Ova interdisciplinarna tehnička oblast bazira se na primeni raznih inženjerskih znanja i veština iz oblasti biologije, hemije i medicine, u cilju produžavanja i održavanja kvaliteta zdravlja ljudi. Medicinska disciplina koja je od posebnog značaja za biomedicinsko inženjerstvo je elektrofiziologija. U okviru ove oblasti vrše se istraživanja električne aktivnosti ćelija u živim organizmima, kao i procesa prenošenja ove aktivnosti na ćelijskom i molekularnom nivou. Ova aktivnost može se meriti i tretirati kao bilo koji električni signal [2].

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Darko Stanišić, docent.

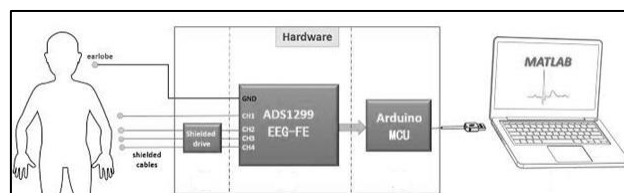
2. OSNOVE ELEKTROFIZIOLOŠKOG MERENJA**2.1. Elektrofiziologija i bioelektrični signali**

Određene ćelije u živim organizmima imaju sposobnost generisanja električnih signala koji se nadalje provode kroz tkivo. Ove vrste ćelija nazivaju se ekscitabilnim ćelijama i u njih spadaju: neuroni, mišićne i srčane ćelije. Najzastupljeniji bioelektrični signali su: elektrokardiogram (EKG), elektromiogram (EMG) i elektroencefalogram (EEG).

Njihova analiza ima veliku ulogu u medicinskoj dijagnostici jer se na osnovu njih mogu izvući mnogi zaključci u vezi aktivnosti srca, mozga i mišića [3]. EKG signali se mogu aproksimirati kao periodični. Imaju frekvencijski opseg od 0.05 Hz do 100 Hz dok je amplitudski opseg od 0.1 – 5 mV. EEG i EMG signali su za razliku od EKG-a stohastičke prirode te je njihova analiza kompleksnija. Frekvencijski i amplitudski opseg EEG-a su 0.1 – 100 Hz i 0.025 – 0.1 mV, respektivno; dok su ove vrednosti za EMG 0 - 500 Hz za frekvencijski i 0.1 – 10 mV za amplitudski opseg [4].

2.2. Elektrofiziološko merenje i instrumentacija

Prema radu istraživača Yanbing Jianga [5], jedan elektrofiziološki akvizicioni sistem (Slika 1) mogao bi da se sastoji iz nekoliko celina. Ovakva implementacija je jedna od mogućih i njeni delovi mogu se razlikovati.



Slika 1. Shematski prikaz jednog elektrofiziološkog sistema

Prvu celinu predstavljaju elektrode, pretvarači jonske struje organizma u struju slobodnih elektrona u provodniku [6].

Drugu celinu predstavlja integrisano kolo, u ovom slučaju ADS1299 EEG-FE proizvođača Texas Instruments. Jedna od bitnih komponenti integrisanih kola koja se proizvode za elektrofiziološke sisteme su instrumentacioni pojačavači. Naime, naponski signali koji se dobijaju preko elektroda imaju nisku amplitudu i dominantan šum te mora postojati element koji ovaj šum smanjuje dok signal od interesa, bioelektrični signal, pojačava toliko da on postaje dominantan u odnosu na šum. U tom cilju se koriste instrumentacioni pojačavači [6].

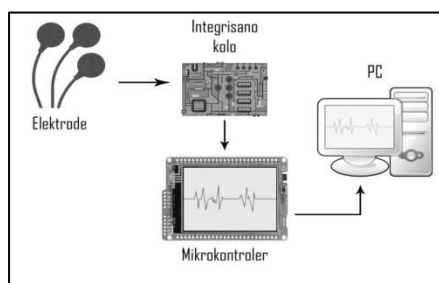
Integrirano kolo takođe vrši analognu – digitalnu (A/D) konverziju signala. Kako bi se signal analizirao i procesirao na digitalnom uređaju, neophodno je prvo da se prevede u digitalni oblik. Zato ova integrirana kola poseduju niz A/D konvertora.

Treću celinu sistema predstavlja digitalni uređaj koji služi za prikupljanje i obradu podataka koje dobija od A/D konvertora. To je najčešće mikrokontrolerski uređaj. U svom projektu, Yanbing Jiang koristio je Arduino razvojno okruženje.

Poslednju celinu predstavlja računar. Podaci se šalju sa mikrokontrolera određenim vidom komunikacije, u ovom slučaju USB serijskom komunikacijom, na računar gde sledi dalja obrada i prikaz, kao i analiza signala [5].

3. OPIS REALIZOVANOG SISTEMA

Realizovani elektrofiziološki sistem (Slika 2) sastoji se iz četiri celine: elektroda, integrisanog kola ADS1299, mikrokontrolera ATxmega128A1 i aplikacije na računaru.



Slika 2. Shematski prikaz realizovanog elektrofiziološkog sistema

Osam elektroda povezano je na integrirano kolo preko DB25 (engl. *DataBus*) konektora. ADS1299, proizvođača *Texas Instruments* spada u grupu integriranih kola koja se mogu koristiti za elektrofiziološka merenja. Neke od istaknutih specifikacija ovog integriranog kola su:

- 8 analognih ulaza sa filtrima elektromagnetnih interferencija
- Multiplexer na ulazu
- 24-bitni analognu–digitalni konvertori
- programabilni pojačavači
- interfejs za serijsku komunikaciju [7].

Mikrokontroler *ATxmega128A1*, proizvođača *Atmel*, spada u grupu *AVR XMEGA* kontrolera. U njegovom sastavu nalazi se 8 – bitni mikroprocesor RISC (engl. *Reduced Instruction Set Computer*) arhitekture. U njemu je integrirano 128 kilobajta programske ili *flash* memorije, 8 kilobajta SRAM(engl. *Static Random – Access Memory*) memorije i 2048 bajtova EEPROM(engl. *Electrically Erasable Programmable Read-Only Memory*) memorije. Neki od bitnijih modula ovog uređaja su: ulazno – izlazni portovi opšte i specifične namene, osam 16 – bitnih tajmera i brojača i modul za serijsku komunikaciju [8]. Ovaj mikrokontroler ulazi u sastav razvojnog multimedijalnog kompleta *Mikromedia for Xmega*. Pored mikrokontrolerskog uređaja u sastav ovog kompleta ulaze:

- TFT (engl. *Thin – Film – Transistor*) displej tipa LCD (*Liquid Crystal Display*), koji je osetljiv na dodir, sa rezolucijom od 320x240 piksela
- Čitač za SD (engl. *Secure Digital*) karticu
- Konektor za USB (engl. *Universal Serial Bus*) tipa mini-B
- Baterijsko i direktno napajanje sistema [9].

3. RAZVIJENA RAČUNARSKA APLIKACIJA ZA ELEKTROFIZIOLOŠKI SISTEM

Računarska aplikacija za predstavljeni elektrofiziološki sistem razvijana je u *Microsoft Visual studio* okruženju, verziji 2019. Programski jezik koji je bio korišćen je C# (engl. *C sharp*), dok je tip aplikacije *Windows form application*. Cilj aplikacije je da se preko nje preuzmu sirovi podaci sa mikrokontrolera putem USB komunikacije, izvrši njihovu obradu i prikaz. U okviru aplikacije razvijene su sledeće funkcionalnosti:

- Kreiranje i izmena korisničkih naloga
- Podešavanje snimanja
- Učitavanje i čuvanje elektrofiziološkog snimka u direktorijumu korisnika
- Izvršavanje novog elektrofiziološkog snimanja

3.1. Korisnički interfejs aplikacije

Grafički interfejs sastoji se od nekoliko obrazaca ili *Windows Form*-i. Nakon inicijalnog obrasca, *WelcomeScreen_Form* pojavljuje se obrazac glavnog menija. U njemu, korisnik mora korak po korak da odabere opcije kako bi došao do sledećeg obrasca. Preostali obrasci u aplikaciji su :

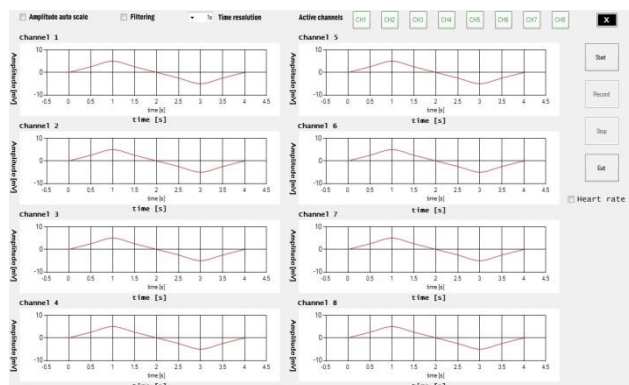
- *New_patient_Form* – kreiranje novog korisnika
- *Show_recording_Form* – prikaz postojećih snimaka
- *USB_Form* – povezivanje na određeni komunikacioni ulaz
- *Impedance_recording* – snimanje impedanse elektroda
- *EKG_Form* – snimanje EKG signala
- *Signal_record_Form* – druga snimanja elektrofizioloških signala

Korisnik ima mogućnost kreiranja novog naloga ili korišćenja postojećeg. Prilikom kreiranja novog naloga, neophodno je uneti bitne podatke o klijentu.

Nakon unošenja podataka, formira se tekstualna datoteka u kojoj se podaci čuvaju. Ukoliko se uvozi postojeći nalog, neophodno je pronaći i odabrati ovu tekstualnu datoteku.

Kada se izvrši snimanje, u direktorijumu trenutnog klijenta formira se novi direktorijum i tekstualna datoteka sa podacima snimanja. Podaci u datoteci su raspoređeni tako da prva kolona predstavlja broj odbirka signala, dok ostale kolone predstavljaju vrednosti amplitude nefiltriranog signala. Nakon kreiranja naloga ili

prijavljivanja na postojeći, korisnik može da bira da izvrši novo snimanje. Računarska aplikacija je povezana sa mikrokontrolerom preko USB komunikacije. Kada se aplikacija poveže sa uređajem, moguće je preći u obrazac za snimanje signala (Slika 3). U njemu je moguće menjati parametre snimanja kao što su: podešavanje vremenske rezolucije, uključivanje ili isključivanje filtriranja signala, skaliranje amplitudske ose, uključivanje ili isključivanje pojedinačnih kanala.



Slika 3. Obrazac za snimanje signala

3.1. Programska realizacija aplikacije

U programskoj realizaciji aplikacije korišćena je *Windows Form* platforma. Ona je zasnovana na kreiranju grafičkih elemenata, obrazaca ili *form*-i sa kojima korisnik vrši interakciju.

Najbitnije funkcionalne celine aplikacije predstavljaju *USB* komunikacija sa mikrokontrolerom, radnje nad podacima kao što su: primanje, obrada, prikazivanje i čuvanje, kao i obrada dobijenih signala. Proces povezivanja aplikacije i mikrokontrolera počinje kada korisnik u glavnom meniju odabere opciju da se započne komunikacija i kada odabere odgovarajući komunikacioni ulaz, *COMPORT* (engl. *CommunicationPort*). Nakon toga može da se pokrene snimanje preko serijske veze. Svaki put kada se podaci sa serijskog ulaza prime, uključuje se programski prekid ulaza, *Port_DataReceived*. U okviru ovog prekida radi se preuzimanje i obrada podataka. Podaci od interesa su poruke. Kako bi se sadržaj nepotpunih odnosno nedovršenih poruka sačuvao, bilo je neophodno formirati pomoćnu kružnu memoriju, odnosno kružni bafer.

Kada se poruka izvuče iz bafera, ona se obrađuje. Poruka sadrži vrednosti signala sa svih osam kanala. Neophodno je izvršiti pretvaranje vrednosti svakog kanala iz formata 3 bajta u decimalnu, označenu vrednost. Integrisano kolo *ADS1299* šalje podatke za jedan kanal u vidu 3 bajta u formatu komplementa dvojke koji služi za binarno predstavljanje označenih brojeva. Opseg vrednosti kola je 2^{24} . Polovinu ovog opsega predstavljaju pozitivne, dok drugu polovinu predstavljaju negativne vrednosti. Najmanje promena vrednosti napona koju kolo može da detektuje definisana je izrazom [7]:

$$\frac{2 \times V_{ref}}{2^{24} \times Gain} \quad (1)$$

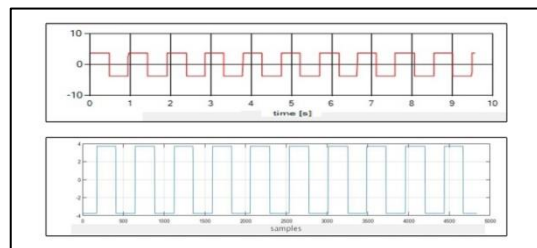
V_{ref} je referentni napon kola koji iznosi 4.5V, $Gain$ predstavlja pojačanje kola koje je podešeno na 24x.

Nakon uspešne konverzije podataka aktivira se događaj preko kojeg se podaci šalju u obrazac za njihov prikaz. Kako bi se prikazala stvarna vrednost signala neophodno ih je prvo filtrirati. Elektrofiziološki signali obično imaju pomerenu baznu liniju i prisutan šum električne mreže. Zbog toga su projektovani filtri za uklanjanje ovih komponenti. Pomeranje bazne linije nastaje usled postojanja jednosmerne komponente te je za njeno otklanjanje projektovan filter propusnik visokih učestanosti. Šum električne mreže javlja se na frekvenciji mreže od 50 herca. Za otklanjanje šuma projektovan je *notch* filter koji otklanja komponentu određene frekvencije iz signala [10].

Filtri su projektovani u Matlab okruženju kako bi se izračunali njihovi koeficijenti koji su se kasnije koristili u *Visual Studio* okruženju.

4. PROVERA REALIZOVANE APLIKACIJE

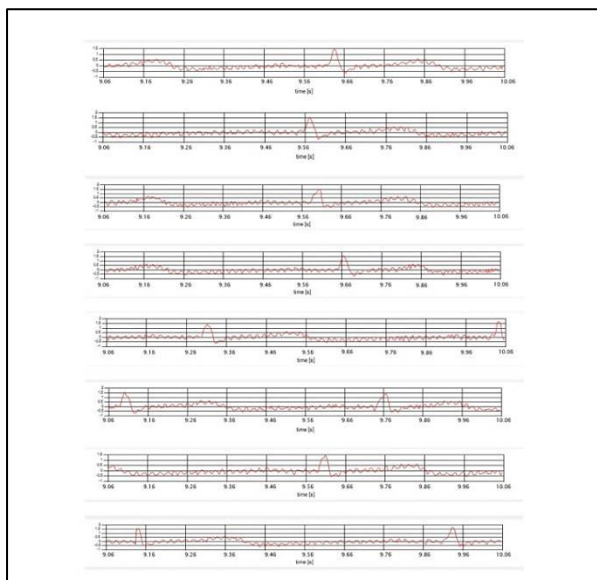
Prvu fazu verifikacije aplikacije predstavlja provera serijske komunikacije. Korišćen je periodičan pravougaoni test signal, sa amplitudom ± 3.75 mV i periodom od jedne sekunde. Na Slici 4 prikazan je test signal u obrascu za prikaz snimanja kao i u Matlab programskom okruženju u cilju utvrđivanja validnosti podataka. Takođe je bilo uključeno vremensko skaliranje na 10 sekundi kako bi se signal mogao bolje videti.



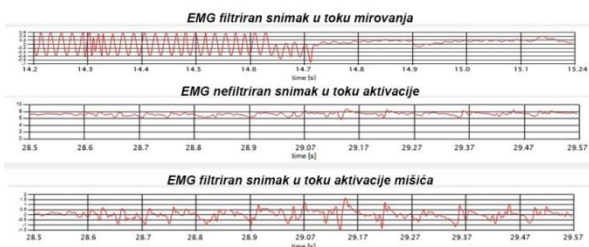
Slika 4. Test signal u aplikaciji i u Matlab okruženju

U drugoj fazi izvršeno je EKG snimanje svih kanala i EMG snimanje jednog kanala i korišćeno je amplitudsko skaliranje kao i filtriranje signala. EKG signali na Slici 5 propušteni su kroz filter za otklanjanje bazne linije i kroz filter za uklanjanje šuma mreže.

EMG signal na Slici 6 prikazan je u mirovanju i prilikom aktivacije mišića. Takođe je prikazan filtriran i nefiltriran signal kada je aktiviran mišić. Skaliranje amplitude nije bilo uključeno prilikom prikazivanja nefiltriranog signala kako bi se videla pomerena bazna linija.



Slika 5. Filtrirani EKG signali sa svih osam kanala prikazani u aplikaciji



Slika 6. EMG signali prikazani u aplikaciji

5. ZAKLJUČAK

U ovom radu predstavljen je jedan edukativni sistem i računarska aplikacija koji bi trebali da služe kako predavačima da lakše objasne materiju, tako i studentima da je bolje savladaju. Aplikacija se može primeniti za objašnjavanje elektrofizioloških pojava i obradu signala dobijenih snimanjem, kao i za ilustrovanje toga kako se analizom dobijenih rezultata mogu izvesti zaključci iz oblasti medicine. Studenti bi imali mogućnost da vide kako se vrši jedno elektrofiziološko snimanje i da ga sami izvrše.

6. LITERATURA

- [1] Richey, R. C., Silber, K. H., Ely, D. P. , „*Reflections on the 2008 AECT Definitions of the Field*“, *TechTrends*, 52(1), pp. 24-25. , 2008
- [2] Matt Carter, Jennifer Shieh, „*Guide to Research Techniques in Neuroscience (Second Edition)*“, Academic Press, 2015
- [3] http://www.fasper.bg.ac.rs/do2016/nastavnici/Milovanovic_Maja/predavanja/4-%20Bioelektricni%20fenomeni%20ekscitabilnih%20celija.pdf (pristupljeno u avgustu 2019.)
- [4] <http://mx.nthu.edu.tw/~yucsu/3271/p07.pdf> (pristupljeno u avgustu 2019.)
- [5] <https://pdfs.semanticscholar.org/8bc0/c375c65371b747dc43c8366815ec3cde16eb.pdf> (pristupljeno u avgustu 2019.)
- [6] http://www.automatika.ftn.uns.ac.rs/images/predmeti/Osnove%20biomedicinskog%20inzenjerstva/Predavanja/Elektrode_3.pdf (pristupljeno u avgustu 2019.)
- [7] <http://www.ti.com/lit/ds/symlink/ads1299.pdf> (pristupljeno u avgustu 2019.)
- [8] http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-8067-8-and-16-bit-AVR-Microcontrollers-ATxmega64A1-ATxmega128A1_Datasheet.pdf (pristupljeno u avgustu 2019.)
- [9] <https://download.mikroe.com/documents/smart-displays/mikromedia/3/xmega/mikromedia-xmega-manual-v111b.pdf> (pristupljeno u avgustu 2019.)
- [10] <http://www.jscholaronline.org/articles/JBER/Sigal-Processing.pdf> (pristupljeno u avgustu 2019.)

Kratka biografija:

Aleksa Mitreski rođen je u Novom Sadu 1995. godine. Gimnaziju „Svetozar Marković“ u Novom Sadu završava 2014. godine i potom upisuje osnovne studije na Fakultetu tehničkih nauka, smer Računarstvo i automatika. Osnovne studije završava 2018. godine i iste godine upisuje master studije. Master rad na Fakultetu tehničkih nauka u Novom Sadu, iz oblasti Biomedicinskog inženjerstva, odbranio je 2019. godine. Kontakt: aleksa_mitreski@uns.ac.rs

PREDVIĐANJE EFEKATA TEMPORALNOG FILTRA NA KVALITET VIDEO SEKVENCI**PREDICTING TEMPORAL FILTER EFFECTS ON QUALITY OF VIDEO SEQUENCES**

Miloš Kudus, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu je opisan algoritam mašinskog učenja za predviđanje efekata temporalnog filtra na video sekvence.

Ključne reči: Kvalitet videa, Obrada slike, Mašinsko učenje

Abstract – This paper describes a machine learning algorithm for predicting temporal filter effects on video sequences.

Keywords: Video quality, Image processing, Machine learning

1. UVOD

Cilj rada jeste kreiranje algoritma za predviđanje efekata temporalnog filtra na video sekvence. Efekat se evaluira na osnovu promene mera šuma i oštine. Nakon primene temporalnog filtra mera šuma videa se popravljja, dok se mera oštine kvira. Amplituda promene mera zavisi od videa i od parametara filtriranja.

Generalno, sekvence koje imaju jak temporalni šum i koje su uglavnom statične imaju najviše benefita od obrade. Takvim videima se mera šuma popravljja značajno dok se mera oštine kvira za veoma male vrednosti.

Na dinamičnim sekvencama sa mnogo kretanja i sa malo šuma je kontraproduktivno primenjivati filtriranje jer je mera šuma već poprilično dobra i nema potrebe za poboljšanjem, a oština objekata koji se kreću se gubi u velikoj meri.

Za predviđanje efekata obrade je korišćena konvoluciona neuronska mreža za regresiju, čiji je izlaz promena kvaliteta video sekvence nakon obrade temporalnim filtrom. Kvalitet video sekvence se računa kao zbir mera kvaliteta šuma. Ulazi su četiri uzastopna frejma monohromatske sekvence. Predobrada je vršena u MATLAB-u, a neuronska mreža je implementirana u Python-u, pri čemu su korišćene biblioteke: *tensorflow* [4][5], *numpy*, *os*, *matplotlib*, *math*, *pandas*, *opencv*.

2. BAZE PODATAKA**2.1. Baza 1**

Prvu bazu čini GMM baza fluoroskopskih videa. Sastoji se od 10 videa različite rezolucije i trajanja. Sekvence su snimljene digitalnim fluoroskopom.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Petrović, vanr. prof.

Iz smanjenog skupa izabrano je nasumično 25% sekvenci za test skup (ukupno 3 sekvence), a ostalih 75% je korišćeno kao trening skup (ukupno 7 video sekvenci).

2.2. Baza 2

Drugu bazu čini eVQA Video Dataset baza [3] koja je sakupljena godinama radom na eVQA projekatima. Sastoji se od dva podskupa (sirovi i izvedeni). Za ovaj zadatak je korišćen deo sirovog podskupa (njih 134 od 489).

Video sekvence su različite rezolucije i trajanja. Snimane su kamerama sa senzorima vidljive svetlosti (monohrom i u boji), IR spektra (LWIR, MWIR, SWIR), ili Visible intensified senzorima.

Sekvence su birane tako da u skupu ne bude više sekvenci koje prikazuju istu scenu, snimanu istim tipom senzora kako ne bi došlo do overfittinga na te video sekvence. Iz tog smanjenog skupa izabrano je nasumično 25% sekvenci za test skup (ukupno 34 sekvence), a ostalih 75% je korišćeno kao trening skup (ukupno 100 video sekvenci).

3. PREDOBRAĐA

Neuronska mreža zahteva ulazne podatke fiksne dimenzije [2]. To predstavlja problem, jer su video sekvence različite rezolucije i trajanja, sa različitim brojem kanala. Problem različitog broja kanala je rešen tako što su sve sekvence prebačene u monohromatske usrednjavanjem sekvenci po kanalima. Što se tiče različitog trajanja, problem je rešen tako što se mreži prosleđuju samo po četiri uzastopne sekvence. Time je smanjena računaska složenost modela jer se izbegla upotreba rekurentnih slojeva.

Problem različitih rezolucija mogao je biti rešen na različite načine.

Prvi način bi mogao da bude proširivanje sekvenci nulama kako bi svi videi imali iste prostorne dimenzije. Taj pristup je odbačen jer računski usložnjava problem.

Drugi način bi bio redimenzionisanje frejmova na neku fiksnu veličinu ali je to odbačeno zbog toga što bi to moglo uticati na mere šuma i oštine. Treći pristup, koji je i korišćen u radu, jeste da se iseke delovi frejma fiksne dimenzije (192x192).

Time se smanjila računaska složenost jer su uzorci koji ulaze u mrežu manjih dimenzija, a šum i oština nisu izmenjeni redimenzionisanjem. Dakle, uzorak za neuronsku mrežu se dobija isecanjem regiona 192x192 iz četiri uzastopna frejma i njihovim prebacivanjem u

monohromatsku sliku. Nakon toga evaluirane su mere šuma i oštine za taj uzorak. Potom je uzorak filtriran temporalnim filtrom i nakon toga su ponovo evaluirane mere šuma i oštine za taj uzorak čime su dobijene mere šuma i oštine nakon obrade temporalnim filtrom.

S obzirom da su RGB kanali uzoraka zapravo različiti frejmovi u vremenu, statični objekti su sivi jer je vrednost na pikselima sva tri kanala podjednaka. Kada imamo kretanje ili šum, vrednosti piksela se menjaju iz frejma u frejm što na ovim slikama vidimo kao boju.

4. UKLANJANJE TEMPORALNOG ŠUMA

Za uklanjanje temporalnog šuma korišćen je temporalni Gausov filtar. Obradeni frejm se dobija kao linearna kombinacija prethodnih frejmova (1).

Koeficijenti kojima frejmovi doprinose obrađenom frejmu su odbirci Gausove funkcije sa centrom u trenutnom frejmu i jediničnom standardnom devijacijom.

$$I_{obr}(i, j, n) = \sum_{k=0}^2 I(i, j, n - k) * TF(n - k) \quad (1)$$

I_{obr} predstavlja piksel obrađene sekvence, i i j su prostorne koordinate, a n je diskretno vreme. I je originalna video sekvenca, a TF su koeficijenti temporalnog filtra.

Direktna posledica uzimanja odbiraka Gausove funkcije za težinske koeficijente jeste ta da što je kadar vremenski udaljeniji od trenutnog kadra, manje doprinosi obrađenom kadru.

Radi smanjenja računске složenosti u obzir su uzeti samo trenutni i dva prethodna kadra koji doprinose sa oko 95% obrađenom kadru. Ovaj filtar smanjuje temporalni šum, ali objekti koji se kreću izgledaju zamućeno.

Za GMM skup srednja vrednost mere šuma posle obrade je 0.97, a standardna devijacija je 0.07. Srednja vrednost mere oštine posle obrade je 0.98, a standardna devijacija je 0.06. Za eVQA skup srednja vrednost mere šuma posle obrade je 0.95, a standardna devijacija je 0.11. Srednja vrednost mere oštine posle obrade je 0.63, a standardna devijacija je 0.26.

Na osnovu srednjih vrednosti mera vidimo da temporalni filtar uglavnom kviri meru oštine (vrednost mere oštine opada), ali istovremeno uglavnom popravlja meru šuma (vrednost mere šuma raste).

Kao konačna vrednost koju mreža treba da predvidi korišćena je promena kvaliteta sekvence nakon filtracije. Ona je data sa (2):

$$\Delta \text{kvalitet} = NMB + SMB - NMA - SMA \quad (2)$$

NMB i SMB predstavljaju meru šuma i oštine pre obrade, respektivno. NMA i SMA predstavljaju meru šuma i oštine nakon obrade, respektivno.

6. OBJEKTIVNE MERE

6.1. Mera šuma

Mera šuma je udeo frejma koji nema nestruktuisane nasumične varijacije između trenutnog i prethodnog

frejma [3]. Što je površina zahvaćena temporalnim šumom veća to je vrednost mere šuma manja. Opseg joj je od 0 do 1. Vrednost 1 znači da temporalni šum nije uočljiv u sekvenci, a vrednost 0 znači da je da je cela površina frejma prekrivena temporalnim šumom.

6.2. Mera oštine

Mera oštine je relativna snaga na visokoj frekvenciji u odnosu na niske frekvencije na lokacijama na kojima je najveća razlika između trenutnog i prethodnog frejma [3]. Idealno oštre slike imaju veću energiju na višim frekvencijama nego na nižim frekvencijama.

Više vrednosti ove mere znače da su u frejmu strukture oštih ivica, niske vrednosti ove mere znače da je slika zamućena. Opseg vrednosti joj je od 0 do 1. Vrednost mere 1 znači da su ivice struktura u frejmu idealno oštre, dok vrednost 0 znači da je frejm jako zamućen, tako da je energija na najvišim frekvencijama spektra 0.

7. NEURONSKA MREŽA

7.1. Objektivne funkcije

7.1.1. L2 norma

L2 ili Euklidska norma je jedna od najčešće korišćenih normi. Penalizuje velika odstupanja vrednosti (3).

$$\text{loss} = \frac{1}{\text{broj uzoraka}} \sum_{\text{uzorci}} \sqrt{\sum_{\text{izlazi}} (y - \hat{y})^2} \quad (3)$$

7.1.2. Tačnost klasifikacije

Tačnost klasifikacije se definiše kao procenat tačno klasifikovanih uzoraka. Da bi problem regresije pretvorili u problem klasifikacije, bilo je potrebno odrediti granicu promene kvaliteta za koju je potrebno vršiti obradu. U ovom radu je za taj prag odabrana vrednost 0,05. Dakle, sve ocene koje su veće od 0,05 su smatrane kao jedna klasa, a ostale kao druga klasa.

7.2. Optimizacija – Adam Optimizer

Adaptive moment estimation (Adam) je algoritam za efikasnu optimizaciju težina i ofseta u neuronskim mrežama [6]. Pokazao se dobro u praksi i široko je korišćen.

Taj algoritam poseduje dobre osobine metode opadajućih gradijenata sa inercijom i *RMS prop*-a. Algoritam prvo izračunava gradijente $\partial \omega$ i ∂b na beču na isti način kao što se to radi metodom opadanja gradijenata.

Beč je grupa uzoraka koji zajedno ulaze u mrežu. Potom se računaju vrednosti gradijenata sa inercijom $V\partial \omega$ i $V\partial b$. Zatim se računa *RMS* ažuriranje $S\partial \omega$ i $S\partial b$. Nakon toga se računaju korekcije tih vrednosti. I konačno, na osnovu toga ažuriramo vrednosti ω i b .

7.3. Trening

Za trening se koriste bečevi sa po jednim uzorkom iz svake trening sekvence. Pošto ima 7 (odnosno 100 za eVQA skup) trening sekvenci, svaki trening beč ima 7

(odnosno 100 za eVQA skup) uzoraka. Redosled kojim uzorci ulaze u mrežu se određuje posebno za svaki video. Redosled kojim uzorci jednog videa ulaze u mrežu je nasumičan sa najmanjim mogućim brojem ponavljanja nekog uzorka.

Permutuje se redosled kojim su izvučeni uzorci iz videa nasumično. Mreža se trenira sa maksimalno 5000 bečeva. Trening prekidamo i ako tačnost na test skupu počne značajno da opada. Za optimizaciju se koristi Adam algoritam i optimizuje se srednja L2 norma, ali računa se i L1 norma i tačnost klasifikacije. Brzina učenja opada linearno sa iteracijom.

7.4. Test

Za test se koriste bečevi sa po 90 (odnosno 10 za eVQA skup) uzoraka iz svake test sekvence. Pošto ima 3 (odnosno 34 za eVQA skup) test sekvenci svaki test, beč ima 270 (odnosno 340 za eVQA skup) uzoraka. Redosled kojim uzorci jednog videa ulaze u mrežu je nasumičan sa najmanjim mogućim brojem ponavljanja nekog uzorka. Permutuje se redosled kojim se izvlače uzorci iz videa nasumično.

Evaluacija jednog test beča se vrši na svakih 50 (odnosno 5 za eVQA skup) evaluacija treninga. Mere se tačnost klasifikacije, L1 i L2 norma. Mreža se snima ukoliko je vrednost tačnosti u trenutnom test beču veća od tačnosti na prethodnim test bečevima.

7.5. Opadanje brzine učenja

U prvim iteracijama treninga objektivna funkcija brzo opada. Kako iteracije odmiču, to opadanje usporava, da bi na kraju počelo da osciluje oko neke vrednosti. Objektivna funkcija ne može dalje da opada jer mreža pravi prevelike korake ažuriranja.

Bilo bi korisno da se brzina učenja smanjuje tokom vremena kako bi mreža, kada se približi lokalnom minimumu, pravila manje korake i time sprečila objektivnu funkciju da osciluje.

U ovom radu je implementirano linearno opadanje brzine učenja zbog svoje jednostavnosti [6]. Jednačina po kojoj brzina učenja opada je data sa (4). Na taj način brzina učenja se kontroliše sa 2 parametra: LRstart i LREndDecay.

$$LR = \frac{(LRstart - LREndDec)}{(maks. br. tr bečeva - 1)}it + LRstart \quad (4)$$

LR je brzina učenja, LRstart brzina učenja u nultoj iteraciji i LREndDec brzina učenja u iteraciji u kojoj želimo da zaustavimo opadanje brzine učenja, it je iteracija. Nakon te iteracije brzina učenja LREndDecay se zadržava do kraja obuke.

7.6. Uslovi za prekidanje treninga

Ukoliko tačnost klasifikacije videa na one koje je poželjno obraditi i na one koje je nepoželjno obraditi na test skupu počne da opada, to je znak overfittinga. Pošto je evaluacija tačnosti na test beču sklona šumu zbog konačnog broja test uzoraka koji se odabira nasumično iz

većeg test skupa, ne bi bilo dobro prekidati obuku na prvom test beču koji je imao manju tačnost od prethodnog. Umesto toga se čeka da trend opadanja tačnosti bude evidentan u više uzastopnih evaluacija na test bečevima. Obuka se prekida ako je gradijent tačnosti na prethodnih n test evaluacija negativan, ili ako je u više gradijenata u prethodnih n iteracija bilo negativno nego pozitivno.

8. REZULTATI

Vršene su obuke sa različitim parametrima sa različitim skupovima. Mreža je snimana u trenucima kada je zabeležena najveća tačnost na test beču do tog trenutka. U tabeli 8.1 su dati rezultati evaluacija na test skupu u nedostatku dovoljnog broja videa za podelu podataka u tri skupa. Podati su dati za GMM skup.

Tabela 8.1. Rezultati neuronske mreže

Mreža	1	2	3	4
Skup	GMM	GMM	GMM	eVQA
DKF	1	1	3	3
KS	32 16 8	32	32	32
MP	2 2 8	32	32	32
PPS	8 2 1	8 1	8 1	8 1
ACT	<i>LeakyReLU</i>	<i>LeakyReLU</i>	<i>LeakyReLU</i>	<i>LeakyReLU</i>
ACC	95.2%	95.2%	96.3%	71.2%
FPR	8.3%	1.3%	1.4%	23%
TPR	96%	90.6%	93.5%	63.5%

Sa DKF je označena dimenzija konvolucionih filtera; sa KS broj filtera po konvolucionom sloju; MP predstavlja dimenzije Max-Pooling prozora po slojevima; PPS označava broj potpuno povezanih neurona po sloju; ACT stoji za aktivacionu funkciju; ACC predstavlja tačnost (engl. Accuracy); FPR je stopa lažnih pozitivna (engl. False Positive Rate) i TPR stopa pravih pozitivna (engl. True Positive Rate).

9. ZAKLJUČAK

Iz rezultata vidimo da su mreže na GMM skupu ostvarile zadovoljavajući učinak. Najbolju tačnost od 96.3% je dala mreža s rednim brojem 3: sa jednim konvolucionim slojem sa 32 filtra dimenzija 3 x 3 i jednim skrivenim potpuno povezanim slojem sa 8 neurona.

Ta mreža je i računski jednostavna, što je bio jedan od zahteva zadatka. Mreža iste takve arhitekture je istrenirana na eVQA skupu, pri čemu je ostvarila mnogo lošiju tačnost od 71.2%.

U daljem radu je potrebno konstruisati mrežu koja daje bolje rezultate i na većem skupu i potencijalno podeliti eVQA skup na više podskupova na osnovu tipa senzora kojim su snimani. To može biti uzrok podbacivanja

10. LITERATURA

- [1] Vladimir Crnojević, “Prepoznavanje oblika za inženjere”, FTN Izdavaštvo, Novi Sad, 2014.
- [2] Josh Patterson, Adam Gibson, “Deep learning- a practitioner's approach“, Sebastopol, CA 2017
- [3] Dr. V. Petrovic, Mr. M. Willcox, Mr. C. B. Jackson, “Augmenting enhanced video quality assessment and enhanced Video quality control-milestone M2 report”, 2019.
- [4] Deeplearning.ai,
<https://www.coursera.org/learn/introduction-tensorflow>
- [5] Deeplearning.ai,
<https://www.coursera.org/learn/convolutional-neural-networks-tensorflow>
- [6] Deeplearning.ai,
<https://www.youtube.com/channel/UCcIXc5mJsHVYTZR1maL5l9w>

Kratka biografija:



Miloš Kudus rođen je u Somboru 1995. god. Bečelor rad na Fakultetu tehničkih nauka iz oblasti Biomedicinsko inženjerstvo odbranio je 2018.god. kontakt: miloskudas@gmail.com

JEZIK ZA GENERISANJE API TESTOVA ZA GraphQL UPITE

A LANGUAGE FOR GENERATING API TESTS FOR GraphQL QUERIES

Milorad Vojnović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu opisan je dizajn i implementacija jezika specifičnog za domen, koji se koristi za opis i generisanje programskog koda namenjenih za API testove za GraphQL upite.

Ključne reči: GraphQL, textX, DSL, WEB testiranje

Abstract – This paper presents design and implementation of a domain-specific language that is used for a description and a generation of GraphQL API tests.

Keywords: GraphQL, textX, DSL, WEB testing

1. UVOD

Ovaj rad obuhvata dizajn i implementaciju jezika specifičnog za domen, koji se koristi za opis i generisanje programskog koda namenjenih za API testove za GraphQL upite.

1.1 Jezici specifični za domen

Jezici specifični za domen - JSD (*Domain specific language* - DSL) su računarski jezici ograničene ekspresivnosti, koji su fokusirani na određeni domen [1]. Oni su optimizovani za određenu klasu problema, koji se naziva domen [2]. Dobro dizajniran JSD može da zameni veliku količinu koda, koja je potencijalno i komplikovana za razumevanje. Takođe, ako se u jeziku koriste termini iz domena, to omogućava mnogo lakšu komunikaciju sa domenskim ekspertima.

Jedan od načina klasifikacije JSD-a je po tome kako su implementirani. JSD možemo da klasifikujemo kao interni ili eksterni, u zavisnosti od toga da li su implementirani preko nekog jezika domaćina ili ne [1].

Interni JSD je jezik koji koristi infrastrukturu postojećeg programskog jezika. Jedan od najpopularnijih internih jezika je *Rails*, koji je implementiran preko *Ruby* programskog jezika. U najviše slučajeva, interni JSD je implementiran kao biblioteka preko postojećeg jezika domaćina.

Eksterni JSD je jezik koji se kreira iz početka i za koji je potrebno definisati posebnu infrastrukturu za parsiranje, interpretiranje, kompajliranje i generisanje koda [3]. Razvijanje eksternog JSD-a je slično implementiranju novog jezika.

Jezici specifični za domen su namenjeni za rešavanje problema iz određenog domena, dok sa druge strane imamo jezike opšte namene - JON (*General purpose language* - GPL), koji se mogu koristiti za bilo koji domen primene [3]. Osnovne razlike između jezika specifičnih za domen i jezika opšte namene mogu se videti u tabeli 1.1 [2].

Tabela 1.1 Razlike između JSD i JON

	JON	JSD
Domen	Velik i kompleksan	Mali i dobro definisan
Veličina jezika	Veliki	Mali
Korisnički definisane apstrakcije	Sofisticirane	Ograničene
Tjuring kompletan	Uvek	Uglavnom ne
Životni vek	Dekade	Godine
Dizajniran od	Guru ili komitet	Par inženjera i domenski ekspert
Korisnička podrška	Velika, anonimna i rasprostranjena	Mala, pristupna i lokalna
Evolucija	Spora, često standardizovana	Brzim tempom
Nevalidne promene (deprecation)	Gotovo nemoguće	Vrlo moguće

Postoji nekoliko razloga zašto koristiti JSD. Jedan od razloga je povećanje produktivnosti programera [1]. JSD-ovi su pogodni za lako razumevanje domena i zato se brže pišu, menjaju i postoji manja verovatnoća za greškom. Drugi razlog je poboljšanje komunikacije između programera i klijenata. Ovo je jako teško postići, ali je benefit veoma velik. Upravo komunikacija između programera i klijenata predstavlja jedan od najvećih problema u razvoju softvera i zbog toga JSD-ovi imaju veliki značaj, jer potencijalno mogu da reše jedan od najvećih problema u razvoju softvera [1].

1.2 GraphQL

GraphQL je upitni jezik za API. Takođe je i radno izvršavanje (*runtime*) za popunjavanje upita konkretnim podacima [4]. On pruža kompletan i razumljiv opis podataka, koje je moguće dobiti sa servera i time pruža *web* klijentima mogućnost da traže minimalni skup potrebnih podataka.

Uz pomoć GraphQL upita je moguće dobiti samo one podatke koji su potrebni. Omogućeno je kreiranje ugnježenih upita, koji omogućava da se povezuju podaci i da se dobavljaju podaci koje bi morali dobiti iz nekoliko HTTP zahteva. Osim toga, moguće je definisati

NAPOMENA:

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

minimalni skup potrebnih podataka, tako da se ne dobija više od onog što je potrebno. Sa *GraphQL*-om je moguće dobiti sve potrebno podatke u jednom zahtevu, ako je šema dobro definisana. Svaki put kada se izvrši upit, on je validiran od strane sistema tipova. Svaki *GraphQL* servis definiše sistem tipova u *GraphQL* šemi preko koje se svi upiti validiraju [4].

Jezik koji se koristi za *GraphQL* zahteve je direktno povezan sa jezikom koji se koristi za odgovore [5]. Ako se analizira JSON odgovor, može se videti da je to rečnik ključeva i vrednosti. Vrednosti dalje mogu biti ugnježdene i sadržati nove parove ključ – vrednost. Ako se posmatraju samo vrednosti iz odgovora, može se videti da su to podaci koji su traženi preko *GraphQL* upita. Na listingu 1.1 se mogu videti primer *GraphQL* upita i JSON odgovora koji prikazuje sličnost zahteva i odgovora [6].

<pre>{ hero { name height mass } }</pre>	<pre>{ "hero": { "name": "Luke Skywalker", "height": 1.72, "mass": 77 } }</pre>
--	---

Listing 1.1 Primeri *GraphQL* zahteva i JSON odgovora

GraphQL menja način dizajniranja API-a [4]. Umesto da se API posmatra kao kolekcija pristupnih tačaka, posmatra se kao kolekcija tipova. Kolekcija tipova se naziva šema. Potrebno je pronaći zajednički jezik preko kog će se opisati šema. Da bi to bilo moguće, *GraphQL* dolazi sa jezikom koji služi za opisivanje šema koji se naziva *Schema Definition Language – SDL* [4]. U *GraphQL* dokumentima se definišu tipovi koji se koriste u šemi i koji će biti dostupni u aplikaciji i koji će biti korišćeni u komunikaciji između klijenta i servera. Tipovi predstavljaju gradivne element *GraphQL* šeme.

1.3 WEB testiranje

U svetu testiranja opšte je poznat pojam piramida testiranja [7]. Na vrhu piramide testiranja se nalaze testovi korisničkog interfejsa (UI-E2E). Ovi testovi prolaze kroz čitavu aplikaciju i ponašaju se kao da korisnik vrši neke akcije u sistemu. Osim ovih testova postoje i integracioni (*integration*) testovi. Oni su poput UI testova, ali oni ne testiraju korisnički interfejs. Umesto toga oni kreću od jednog sloja niže i testiraju servise. Na dnu piramide su jedinični (*unit*) testovi. Oni su mali, precizni i na nivou koda. Oni testiraju malu jedinicu, najčešće klasu i daju precizan izveštaj u tome šta nije zadovoljilo kriterijume testiranja.

Kada se pogleda da ima nekoliko vrsta testova postavlja se pitanje koje testove izabrati u kojoj situaciji. Za ovaj odabir se najčešće koristi pravilo palca (*rules of thumb*) [7]:

- Favorizovati jedinične u odnosu na ostale testove
- Pokriti nedostatke jediničnih testova sa integracionim testovima
- Koristiti UI testove štedljivo

UI i integracioni testovi su za testiranje povezanosti komponenti sistema. Ti testovi su sporiji, jer prolaze kroz više slojeva aplikacije. Sa druge strane jedinični testovi su brzi i daju ranu povratnu informaciju. Oni se koriste najčešće tokom razvoja kako bi što pre dobili informaciju o radu neke komponente. Jedinični testovi se koriste kako bi aplikacija brzo iterirala dok se ostale dve vrste koriste da bi se proverilo da li sve funkcioniše. Obe vrste imaju bitnu namenu.

2. SPECIFIKACIJA SISTEMA

2.1 Pregled stanja u oblasti

Karate je alat otvorenog koda (*open-source*) koji kombinuje automatizaciju testova, kreiranje lažnih objekata (*mocking*) i testove performansi u jedan unificirani okvir [8]. Uz pomoć **Karate**-a je moguće pisati jednostavne testove u čitljivoj sintaksi, koja je pažljivo dizajnirana za HTTP, JSON, *GraphQL* i XML. Osim ovoga, moguće je generisanje odgovarajućih izveštaja o uspešnosti izvršavanja testova. Jedan primer testa napisanog preko **Karate** jezika definisan je u listingu 2.1 [8].

<pre>Scenario Outline: * text query = """ { hero(name: "<name>") { height mass } } """ Given path 'graphql' And request { query: '#{query}' } And header Accept = 'application/json' When method post Then status 200 Examples: name John Smith </pre>
--

Listing 2.1 Test za *GraphQL* API napisan preko **Karate** jezika

Karate je odličan alat koji omogućava testiranje *GraphQL* API-a. JSD preko kog se definišu test slučajevi ima mnoštvo mogućnosti. Jedna od mana jezika jeste to što ovo nije jezik samo za *GraphQL* API, pa zbog toga u samom jeziku ima elemenata koji nisu potrebni.

Just-API je deklarativni okvir za testiranje REST, *GraphQL* ili nekih drugih HTTP baziranih servisa [9]. Moguće je testiranje API-a bez pisanja programskog koda. Test specifikacije se čitaju iz YAML fajlova i mogu da se izvršavaju sekvencijalno ili u paraleli. Takođe je moguće generisanje izveštaja o uspešnosti izvršavanja testova u nekoliko formata uključujući HTML i JSON.

Primer test slučaja za *GraphQL* API napisan preko **Just-API** jezika prikazan je u listingu 2.2 [9]:

```

meta:
  name: GraphQL location service
configuration:
  host: api.graphloc.com
  scheme: https
specs:
  - name: Get Location of a given ip address
    request:
      method: post
      path: /graphql
      headers:
        - name: content-type
          value: application/json
      payload:
        body:
          type: json
          content:
            query: >
              {
                getLoc(ip: "8.8.8.8") {
                  country {
                    iso_code
                  }
                }
              }
            variables: null
            operationName: null
      response:
        status_code: 200
        json_data:
          -path: $.data.getLocation
            value: US

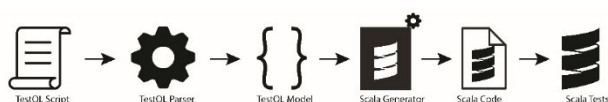
```

Listing 2.2 Test za GraphQL API napisan preko Just-API jezika

Just-API je sličan alat kao i Karate. Ima slične mogućnosti i najveća razlika jeste to što imaju drugačiji JSD za opis testova. Karate koristi specifičnu sintaksu dok se u Just-API-u testovi definišu preko YAML fajlova. S obzirom da se mogu testirati različite vrste API-a, ovaj alat takođe ima manu, jer nije dovoljno specifičan za GraphQL API i da zbog toga ima nepotrebnih stvari u jeziku.

2.2 Arhitektura sistema

TestQL jezik je implementiran kao eksterni JSD. Najpre se napiše TestQL skripta, koja se prosleđuje TestQL parseru. Parser je osnovna komponenta sistema koja na osnovu definisane gramatike parsira skripte koje opisuju test slučajeve. Vrš parsiranje fajlova i proverava da li su programi napisani u skladu sa gramatikom jezika i vrši semantičke provere. Ako je skripta napisana u skladu sa sintaksnim i semantičkim pravilima jezika, kao rezultat se dobila model test slučajeve kao objektna struktura, koja se prosleđuje generatoru test slučajeve. Generator na osnovu datih modela generiše kod za ciljnu platformu, koja je u ovom slučaju Play okvir u Scala programskom jeziku. Na kraju se Scala kod kompajlira i dobijaju se izvršivi testovi. Arhitektura datog sistema je prikazana na slici 2.1.



Slika 2.1 Arhitektura celog TestQL sistema

3. IMPLEMENTACIJA SISTEMA

Gramatika TestQL jezika je opisana u TextX [10] meta-jeziku za opisivanje jezika specifičnih za domen. Na osnovu opisa meta-jezika je izgenerisan parser za TestQL jezik. Generator u okviru TestQL alata je implementiran u programskom jeziku Python, dok je za generisanje koda GraphQL testova korišćen šablon-generator Jinja2.

3.1 Struktura TestQL skripte

Koreni element u TestQL gramatici je *Test* i predstavlja osnovnu strukturu opisa test slučajeve. *Test* element sadrži *Package*, *Class*, *Scenario*, *Request* i jedan ili više *Case* elemenata. *Package* predstavlja naziv paketa u kom će test slučaj biti generisan. *Class* predstavlja naziv klase gde će test slučaj biti generisan. *Scenario* predstavlja opis funkcionalnosti tj. zahteva koji se testira. *Request* predstavlja definiciju GraphQL zahteva koji se treba testirati. *RequestType* definiše tip operacije koji će biti izvršen, i može biti “query” (upit) ili “mutation” (mutacija). *Attribute* definiše strukturu koja se očekuje od servera kao odgovor. *Argument* predstavlja argumente operacije, koji se koriste za slanje podataka na server i filtriranje. *ArgumentValue* predstavlja vrednost argumenta i može biti *SimpleValue*, *CompoundValue* ili *TagValue* element. *SimpleValue* predstavlja jednostavnu vrednost i može biti “STRING”, “INT”, “FLOAT” ili “BOOL”. *CompoundValue* predstavlja složenu vrednost. *TagValue* element predstavlja vrednost koja će biti određena na osnovu vrednosti iz *Examples* elementa. *Case* element predstavlja test slučajeve koji će se testirati. *Response* predstavlja odgovor servera. *JsonObject* predstavlja JSON odgovor servera. *JsonMember* predstavlja atribut u JSON objektu. *JsonValueWithTags* predstavlja vrednosti JSON objekta zajedno sa tag vrednostima i može biti *JsonValue* ili *TagValue*. *JsonValue* predstavlja vrednost u JSON objektu i može biti *JsonObject*, *JsonArray*, *JsonNull* ili *SimpleValue*. *JsonArray* predstavlja JSON niz. *Multiplier* predstavlja element koji umnožava vrednosti u nizu. *MultiplierValue* predstavlja broj za koliko puta će biti umnožen element u nizu i može biti “INT” ili *TagValue* element. *JsonNull* predstavlja null vrednost u JSON objektu. *Examples* predstavlja vrednosti test slučajeve, koji će biti ubačeni umesto *TagValue* vrednosti tokom generisanja. *ExampleAttributes* predstavlja nazive tagova mesto kojih će vrednosti biti ubačene tokom generisanja. *ExampleAttribute* predstavlja konkretan naziv taga. *Example* element predstavlja konkretne vrednosti koje će biti ubačene umesto tagova tokom generisanja. *RowSeparator* predstavlja granicu između zaglavlja tabele i konkretnih vrednosti tabele. *JsonValueWithEmptyValue* predstavlja vrednosti koje će biti ubačene u tagove tokom generisanja i može biti *JsonValue* ili “*”.

Pošto su definisani svi elementi gramatike moguće je definisati kompletne primere u TestQL gramatici tj. *Test* element koji je koreni element gramatike. U listingu 3.1 je prikazan primer *Test* elementa uz pomoć TestQL gramatike, koji opisuje test slučaj za dobavljanje kategorija.

```

package:graphql
class:CategoriesSpec
scenario:retrieving categories

request:
  ``graphql
  query findAllCategories {
    categories {
      id
      name
    }
  }
  ...

response:
  ``json
  {
    "data": {
      "categories": [
        [2]{
          "id": <id>,
          "name": <name>
        }
      ]
    }
  }
  ...

examples:
  | should | id | name |
  |-----|---|-----|
  | "retrieve all categories" | 1 | "RAM memory" |
  | * | 2 | "SSD disk" |

```

Listing 3.1 *TestQL primer Test elementa*

4. ZAKLJUČAK

Razvijeni sistem omogućava generisanje test slučajeva za *GraphQL* API. Generišu se testovi za *Play* okvir za programski jezik *Scala* na osnovu specifikacije definisane u jeziku specifičnom za domen. Uz pomoć jezika je moguće brzo i jednostavno definisati test slučajeve. Domen jezika je ograničen na generisanje test slučajeva za *GraphQL* API, što omogućava da delovi jezika budu isključivo iz tog domena što pruža prednost u odnosu na druge alate, koji su nastali kao proširenje za neki drugi tip API-a.

Moguća primena ovog alata je za pisanje jediničnih ili integracionih testova za *GraphQL* API. TestQL alat je namenjen pre svega za programere. Programeri mogu da ga iskoriste i da brže napišu testove za *GraphQL* API. Domen TestQL alata je testiranje *GraphQL* API-a i upravo to predstavlja najveću prednost ovog alata, jer je domen druga dva testiranja API-a i samim tim njihova konciznost je lošija u odnosu na TestQL.

5. LITERATURA

- [1] M. Fowler, *Domain-Specific Languages*, Addison-Wesley Professional, 2010.
- [2] M. Voelter, *DSL Engineering: Designing, Implementing and Using Domain-Specific Languages*, CreateSpace Independent Publishing Platform, 2013.
- [3] D. Ghosh, *DSLs in action*, Manning Publications Co., 2011.
- [4] A. B. Eve Porcello, *Learning GraphQL: Declarative Data Fetching for Modern Web Apps*, O'Reilly Media, 2018.
- [5] S. Buna, *Learning GraphQL and Relay*, Packt Publishing, 2016.
- [6] „GraphQL,“ Dostupno na mreži: <https://graphql.org/>. [Poslednji pristup 12. 5. 2019.].
- [7] J. Rasmusson, *The Way of the Web Tester: A Beginner's Guide to Automating Tests*, Pragmatic Bookshelf, 2016.
- [8] „Karate,“ Dostupno na mreži: <https://intuit.github.io/karate/>. [Poslednji pristup 1. 6. 2019.].
- [9] „Just-API,“ Dostupno na mreži: <https://kiranz.github.io/just-api/>. [Poslednji pristup 1. 6. 2019.].
- [10] I. Dejanović, „TextX,“ Dostupno na mreži: <http://textx.github.io/textX>. [Poslednji pristup 12. 5. 2019.].

Kratka biografija:

Milorad Vojnović rođen je u Novom Sadu 1994. godine. Master rad na Fakultetu tehničkih nauka u Novom Sadu, iz oblasti Elektrotehnike i računarstva – softversko inženjerstvo, odbranio je 2019. godine.

SISTEM ZA UPRAVLJANJE ŽALBAMA KORISNIKA U DOMENU FINANSIJA CUSTOMER COMPLAINTS MANAGEMENT SYSTEM IN THE FIELD OF FINANCE

Milica Nikolić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – U radu je predstavljen sistem za upravljanje žalbama korisnika u domenu finansija. Sistem je zasnovan nad velikim brojem realnih žalbi sa ciljem da kompaniji pruži mogućnost automatskog upravljanja žalbama, tako što će u realnom vremenu vršiti predikcije koje rešene žalbe ne zadovoljavaju korisnike koji su ih uložili i koji je očekivani broj žalbi za naredni mesec. Krajnji cilj ovog sistema jeste smanjenje broja nezadovoljnih korisnika date kompanije. Zbog rada sa velikom količinom podataka, ceo sistem odvija se u distribuiranom okruženju.

Ključne reči: određivanje sentimenta, klasifikacija žalbe, rad sa velikom količinom podataka, distribuirano okruženje

Abstract – This paper presents the customer complaints management system in the field of finance. The system is based on large number of complaints, with the primary goal of providing the ability of automatic management to the company, by predicting complaints which are likely to be disputed and the number of expected complaints for the following month. The final goal of the system is to reduce the number of unsatisfied customers of the specific company. Since this is a big data problem, the whole system is running in a distributed environment.

Keywords: sentiment mining, complaints classification, big data, distributed environment

1. UVOD

Ulaskom u novu digitalnu eru, započet je proces jednostavnijeg prikupljanja, elektronske razmene i skladištenja informacija.

U cilju zaštite svojih građana, američka vlada osnovala je zavod za finansijsku zaštitu potrošača, čije jedno od odeljenja predstavlja odsek za žalbe korisnika [1]. Svaki građanin, preko veb sajta ovog zavoda može da uloži žalbu na kompaniju sa kojom sarađuje, nakon čega kompanija ima određeni rok da razreši žalbu i da pruži odgovor, koji potom može biti prihvaćen ili osporen od strane korisnika.

Na ovaj način, građani mogu da saznaju koje kompanije su otvorene ka rešavanju problema svojih korisnika, a koje imaju nezadovoljne korisnike. Sa druge strane, kompanije imaju uvid u poslovanje svojih konkurenata.

NAPOMENA:

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

Cilj ovog sistema jeste automatsko upravljanje žalbama korisnika na način da pruži krajnjem korisniku – kompaniji uvid u postojeće žalbe i da bude sposoban da izvrši predikcije intenziteta žalbi za naredni mesec, kao i da uputi kompaniju na one žalbe koje imaju veću mogućnost da budu osporene.

Shodno cilju, sistem je podeljen na dva velika, nezavisna podsistema:

- podsistem za predickiju intenziteta žalbi u narednom mesecu
- podsistem za predikciju odgovora korisnika na razrešenu žalbu – da li će korisnik da prihvati odgovor kompanije ili će da ga ospori.

Ovi podsistemi naslanjaju se na podsistem za prikupljanje i skladištenje podataka.

Predikcija intenziteta žalbi, odnosno broja žalbi u narednom mesecu, zasniva se na upotrebi modela linearne regresije (engl. *Linear Regression*). Kao jedini atribut koristi se broj žalbi iz prethodnog meseca, kako bi se predvideo njihov broj u narednom. Iako predikcije ovog modela nisu naročito precizne, sistem je u stanju da predvidi smer kretanja intenziteta žalbi i da približan opseg broja žalbi u narednom mesecu u veoma kratkom vremenskom periodu, što je od izuzetnog značaja za kompaniju, te, u ovu svrhu, drugi modeli nisu isprobavani.

Za predviđanje osporavanja odgovora kompanije od strane korisnika, isproban je *kernel-based* algoritam - *Support Vector Machine (SVM)*, kao i nekoliko algoritama baziranih na principu stabala odlučivanja. Svi modeli isprobani su nad istim ulaznim skupom podataka koji se sastoji iz sledećih atributa – usluga i problem na koju korisnik ulaže žalbu, odgovor kompanije, atribut koji govori da li kompanija dozvoljava javno objavljivanje odgovora, sentiment teksta žalbe korisnika, atribut koji govori da li korisnik dozvoljava javno objavljivanje tekstualnog opisa žalbe, lokacija sa koje je uložena žalba, način na koji je žalba podneta (usmeno, putem telefona, imejla, itd.), atribut koji govori da li je kompanija pružila odgovor na žalbu u dogovorenom vremenskom roku i atribut koji označava da li je korisnik osporio odgovor kompanije ili ga je prihvatio.

Za implementaciju ovog rešenja, prednost su dobili modeli zasnovani na stablima odlučivanja zbog činjenice da imaju mogućnost da prikažu udeo atributa u konačnoj predikciji, što je od izuzetne važnosti za krajnjeg korisnika (kompaniju).

Najbolje performanse pokazao je *XGBoost (eXtreme Gradient Boosting)* algoritam, što ga je odredilo za konačno rešenje ovog sistema.

2. METODOLOGIJA

Programsko rešenje implementirano je u *Python*-u, pri čemu se ceo proces odvija u distribuiranom okruženju. Za kontrolu rada distribuiranog sistema i za skladištenje podataka, upotrebljena je sistemska platforma *Hadoop* [2], dok se obrada podataka, treniranje modela i dobijanje predikcija vrše pomoću veoma popularnog alata za distribuirano programiranje – *Spark*-a [3]. Logika sistema enkapsulirana je u mikroservis koji putem *REST API*-ja komunicira sa *Angular* [5] veb aplikacijom, preko koje korisnik ostvaruje vezu sa sistemom, izvršava zadatke i dobija rezultate.

Sistem se sastoji iz tri celine. Podsystem za prikupljanje i skladištenje podataka osnova je za rad preostala dva podsystema, koji služe za rešavanje dva glavna problema ovog sistema i čine ih:

- podsystem za predickiju intenziteta žalbi u narednom mesecu
- podsystem za predickiju odgovora korisnika na razrešenu žalbu

2.1. Podsystem za prikupljanje i skladištenje podataka

Podsystem za prikupljanje i skladištenje podataka zadužen je za prikupljanje i skladištenje velike količine podataka, pri čemu se ažuriranje podataka obavlja na dnevnom nivou. Komunikacija i preuzimanje podataka sa zvaničnog sajta [1] vrši se upotrebom *Socrata API*-ja [6] i zasnovana je na slanju *SoQL* (*Socrata Query Language*) upita, jezika koji se u potpunosti oslanja i predstavlja dopunu *SQL*-a (*Structured Query Language*). S obzirom na to da je broj zapisa višemilionski, podaci se skladište na distribuiranom fajl sistemu – *Hadoop Distributed File System* (*HDFS*).

2.2. Podsystem za predickiju intenziteta žalbi u narednom mesecu

Podsystem za predickiju intenziteta, odnosno broja žalbi u narednom mesecu zasnovan je na modelu linearne regresije [7]. Model je izuzetno jednostavan, bez značajnijih meta-parametara, te ne zahteva dodatne pretrage u svrhu traženja najbolje kombinacije njihovih vrednosti.

Sve prikupljene žalbe grupišu se po mesecu i računa se njihov broj, a potom se taj broj i broj žalbi iz narednog meseca (koji predstavlja ciljno obeležje) šalju kao ulaz modelu. Na ovaj način, model se obučava da predvidi broj žalbi u narednom mesecu, ako mu je poznat samo broj žalbi iz trenutnog meseca.

2.3. Podsystem za predickiju odgovora korisnika na razrešenu žalbu

Podsystem za predickiju odgovora korisnika na razrešenu žalbu predstavlja složeniji podsystem i može se posmatrati kao sistem za sebe. Sastoji se iz dva modula:

- podsystem za analizu i obradu podataka
- podsystem za pronalaženje meta-parametara i treniranje modela

2.3.1. Podsystem za analizu i obradu podataka

Podsystem za analizu i obradu podataka kao ulaz dobija višemilionski broj neobrađenih žalbi, vrši njihovo

procesiranje, a potom tako obrađene žalbe šalje na izlaz, odnosno ulaz podsystema za pronalaženje meta-parametara i treniranje modela. Podsystem za analizu i obradu podataka rešava dva velika problema:

- selekcija obeležja
- eksplorativna analiza i priprema ulaznih podataka za treniranje modela.

Model za prikupljanje i skladištenje žalbi skuplja i skladišti žalbe u originalnom formatu, tačno onako kako su objavljene na zvaničnom sajtu [1]. Ovakav skup opisan je pomoću 18 atributa, shodno čemu količina vremena potrebna za njegovu obradu i treniranje modela i procesna moć računara za rad sa ovim podacima rastu. Nad skupovima podataka sa velikim brojem atributa često se vrši selekcija obeležja, pri čemu se znatno smanjuje vreme potrebno za izvršavanje datog problema, što dovodi do toga da je sistem u stanju da pruži odgovor u realnom vremenu.

Selekcija obeležja implementiranog sistema svodi se na dobro poznavanje domena i posmatranje korelacije između različitih atributa. Najpre se iz osnovnog skupa podataka eliminišu atributi poput rednog broja žalbe, oznake koja je dodeljena žalbi i slično, s obzirom na to da ovakvi atributi nemaju uticaj na buduću predickiju. Potom se posmatraju rezultati korelacije između različitih atributa, na koji način dolazi do izbacivanja atributa. Na ovaj način izbačeni su zip kod (koji je u korelaciji sa atributom koji predstavlja lokaciju sa koje je podneta žalba), datum pristizanja i datum prosleđivanja žalbe kompaniji (koji su u korelaciji sa atributom koji govori da li je odgovor na žalbu dat u okviru dogovorenog vremenskog perioda) i još neki slični atributi koji se mogu opisati upotrebom nekih od preostalih atributa skupa podataka. Konačan skup atributa predstavljen je u poglavlju 2.4.

Eksplorativna analiza obuhvata detaljnu analizu, često praćenu grafičkim prikazima, svih atributa ulaznog skupa i svih njihovih mogućih vrednosti. Primenom eksplorativne analize, došlo se do važnih statističkih otkrića poput tih da su svi atributi ulaznog, redukovano skupa kategorički, da se skup sastoji iz velikog procenta nedostajućih vrednosti i da je skup vrednosti ciljne promenljive znatno neizbalansiran.

Binarni kategorički atributi, koji često daju odgovore na da/ne pitanja poput toga da li je žalba osporena, da li je kompanija pružila odgovor u dogovorenom roku, da li je korisnik pristao da se tekst žalbe objavi javno – pretvoreni su u numeričke, prostom dodelom broja 1 kao odgovora “da”, odnosno 0 kao odgovora “ne”, dok su preostali kategorički atributi, čiji broj mogućih vrednosti kategorija prelazi i preko stotinu, pretvoreni u numeričke attribute upotrebom *Label Encoding*-a [8].

Problem nedostajućih vrednosti rešen je popunjavanjem vrednosti atributa pomoću vrednosti nekih drugih atributa, na primer, nedostajuće vrednosti lokacija dobijene su upotrebom vrednosti zip kod atributa. Instance sa velikim brojem nedostajućih vrednosti atributa eliminisane su iz skupa podataka. Jedan od atributa sa najvećim brojem nedostajućih vrednosti jeste upravo sam tekst žalbe, koji bi trebalo da predstavlja glavni atribut prilikom predickije

da li će korisnik prihvatiti odgovor kompanije ili će ga osporiti. Ovaj atribut, predstavljen u tekstualnoj formi, takođe je nepogodan za treniranje izabranog modela, koji ne podržava rad sa tekstualnim podacima, te je atribut zamenjen novim atributom koji predstavlja numeričke vrednosti sentimenta žalbe, dobijene primenom *VADER (Valence Aware Dictionary for sEntiment Reasoning)* [9] algoritma nad datim tekstom. Za vrednosti sentimenta žalbi čiji tekstualni opis nije dat, upotrebljena je vrednost 0, koja označava da tekst ove žalbe nije ni pozitivan ni negativan, s obzirom na to da bi izbacivanje svih instanci sa nedostajućim vrednostima ovog atributa znatno smanjilo skup podataka za treniranje.

Eksplorativna analiza pokazala je i veliki disbalans kada je u pitanju broj osporenih i neosporenih žalbi, tačnije broj neosporenih žalbi daleko je veći od broja osporenih. Za rešavanje ovog problema, primenjena je tehnika *SMOTE (Synthetic Minority Over-sampling Technique)* [10], koja predstavlja jednu od tehnika *oversampling-a*.

2.3.2 Podsystem za pronalaženje meta-parametara i treniranje modela

U svrhu predikcije korisnikovog odgovora na razrešenje žalbe od strane kompanije, iskorišćen je *XGBoost (eXtreme Gradient Boosting)* [11] algoritam. Performanse ovog modela znatno se mogu poboljšati pronalaženjem odgovarajuće kombinacije vrednosti meta-parametara. Za pronalaženje meta-parametara modela koji će rezultirati najboljim performansama, upotrebljena je kombinacija *RandomizedSearchCV* [12] i *GridSearchCV* [13] algoritma.

Postupak traženja ovakvih meta-parametara započinje u-potrebom *RandomizedSearchCV* algoritma koji kao početnu mrežu parametara prima parametre modela sa vrednostima u predefinisanim opsezima.

RandomizedSearchCV vrši nasumičnu pretragu parametara, gde svaka kombinacija predstavlja uzorak distribucije mogućih vrednosti parametara. Po izvršavanju ovog algoritma, dobija se *XGBoost* model sa “najboljim” performansama kao i vrednosti meta-parametara koje su rezultirale ovim modelom. Ovako dobijene vrednosti meta-parametara često ne predstavljaju najbolje već približno najbolje vrednosti, dobijene uz znatno mali utrošak vremena. Potom se, za dobijanje dodatne tačnosti, koristi *GridSearchCV* algoritam, koji se zasniva na isprobavanju svih mogućih vrednosti parametara.

Kao ulazna mreža vrednosti parametara za pretragu koriste se vrednosti dobijene upotrebom *RandomizedSearchCV* algoritma kao i vrednosti iz njihove neposredne okoline. Ovako istreniran model pokazuje najbolje performanse i prihvata se kao konačno rešenje. Predikcije se izvršavaju u realnom vremenu, dok treniranje modela zahteva veću količinu vremena i odvija se u odloženom režimu. Budući da se podaci svakodnevno ažuriraju, model se ponovo trenira na mesečnom nivou.

2.4. Skup ulaznih podataka

Od 18 ulaznih atributa originalnog skupa podataka, 14 atributa ima kategoričke vrednosti, jedan predstavlja tekstualni opis, dok su dva tipa datuma i samo jedan atribut je numerički. Većina kategoričkih atributa velikog

je kardinaliteta – broj mogućih kategorija prelazi preko sto.

Kao ulazni skup podsistema za predikciju intenziteta žalbi u narednom mesecu, uzima se samo datum objavljivanja žalbe na sajtu, tj. datum pristizanja žalbe.

Za pripremu skupa za predikciju odgovora korisnika na razrešenje žalbe, prvo se vrši obrada i analiza podataka koju čine selekcija obeležja i eksplorativna analiza, opisane u poglavlju 2.3.1, nakon čega se skup podataka sastoji iz atributa čiji su opisi dati u tabeli 2.1.

Tabela 2.1: Ulazni skup atributa *XGBoost* modela

ATRIBUT	OPIS ATRIBUTA
product	usluga na koju je žalba upućena
sub_product	pod-kategorija usluge na koju je žalba upućena
issue	problem koji korisnik ima
sub_issue	pod-kategorija problema koji korisnik ima
vader_sentiment	numerička vrednost sentimenta žalbe
company_public_respon se	da li kompanija dozvoljava javno objavljivanje odgovora
state	lokacija sa koje je žalba upućena
consumer_consent_provi ded	da li korisnik dozvoljava javno objavljivanje tekstualnog opisa žalbe
submitted_via	način na koji je žalba upućena zavodu
company_response	odgovor kompanije na žalbu
timely	da li je kompanija pružila odgovor u dogovorenom roku
consumer_disputed	da li je korisnik osporio odgovor kompanije

Ovaj skup atributa čini ulazni skup u *XGBoost* model, pri čemu atribut *consumer_disputed* predstavlja ciljnu promenljivu.

3. EVALUACIJA

Verifikacija celokupnog sistema u suštini se svodi na verifikaciju podsistema za predikciju intenziteta žalbi za naredni mesec i podsistema za predikciju korisnikovog odgovora na razrešenu žalbu od strane kompanije. Tačnije, verifikacija sistema podrazumeva verifikaciju modela linearne regresije i *XGBoost* modela.

Model linearne regresije evaluira se podelom ulaznog skupa na trening i test skup u odnosu 85% na prema 15%, upotrebom vrednosti R^2 mere, koja iznosi 0.64. Iako ovaj model ne daje izuzetno precizna rešenja, empirijskim putem je ustanovljeno da je model u stanju da predvidi smer kretanja intenziteta žalbi u narednom mesecu, tj., da predvidi da li će njihov broj naglo opasti ili porasti, kao i da pruži okviran opseg vrednosti za očekivani broj žalbi u narednom mesecu. Sa druge strane, model je izuzetno jednostavan, bez dodatnih parametara za podešavanje i kao takav ne zahteva veću količinu vremena potrebnu za njegovo treniranje i predikciju rezultata.

Za treniranje *XGBoost* modela koji će pružiti najbolje performanse, vrše se dodatne pretrage, opisane u poglavlju 2.3.2, koje pronalaze takvu kombinaciju vrednosti meta-parametara da istrenirani model rezultira najboljim performansama. Proces pronalaženja ovakvog

modela, sastoji se iz višestrukog treniranja *XGBoost* modela, svaki put sa različitim kombinacijama vrednosti meta-parametara, pri čemu se svaki model evaluira metodom unakrsne evaluacije tako što se ulazni skup deli na k segmenata podeljenih na trening i test skup, a potom se kao konačna tačnost ovog modela uzima prosečna tačnost svih ovih segmenata. Ovakav način evaluacije izabran je zbog činjenice da *RandomizedSearchCV* i *GridSearchCV* algoritam pružaju direktnu podršku za unakrsno evaluiranje, primajući ovu metodu kao jedan od svojih parametara. Kao mere evaluacije koriste se tačnost i odziv. Prvenstveno se traži model sa najvećim odzivom, smatrajući da nekoliko lažno klasifikovanih žalbi ne predstavlja veliku grešku sve dok su sve žalbe koje treba da budu klasifikovane kao osporene - klasifikovane tačno, ali ipak vodeći računa da tačnost modela ne spadne ispod empirijski, unapred određene granice od 70%¹. S obzirom na to da su vrednosti meta-parametara svaki put nasumično odabrane, tačnost ovog modela je promenljiva ali se kreće u neznatno malom opsegu, pri čemu odziv uvek prelazi 85%, dok je tačnost iznad 75%.

Ovaj model pruža mogućnost prikaza udela svakog od pojedinačnih atributa u predikciji. Grafik udela atributa pokazuje da način prilaganja žalbe kao i činjenica da li je kompanija dozvolila javno objavljivanje svog odgovora imaju veliki uticaj na predikciju. Takođe, neočekivano, lokacija sa koje je žalba podneta ima veliku ulogu prilikom predikcije, dok razlozi žalbe i činjenica da li je kompanija odgovorila u zadatom vremenskom roku ne utiču puno na predikciju da li će korisnik osporiti odgovor na žalbu.

Modeli opisani u ovom radu specifično se treniraju nad skupom podataka vezanim za isključivo jednu kompaniju, te i da postoji sistem sličan ovom, verovatno je razvijen u okviru same kompanije i u postojećoj literaturi nije pronađen nijedan rad sa istim ciljem predviđanja, vezan za implementaciju ili rezultate ovakvog sistema, te je teško porediti rezultate sa postojećom literaturom.

4. ZAKLJUČAK

U radu je predstavljen model sistema za upravljanje žalbama korisnika u domenu finansija. Logika celog sistema enkapsulirana je u mikroservis koji putem *REST API*-ja komunicira sa veb aplikacijom pomoću koje krajnji korisnik – kompanija, izvršava određene zadatke i dobija odgovore od sistema. Aplikacija je zasnovana na realnim, javno dostupnim podacima i služi da omogućiti kompaniji uvid u sve do tada njoj upućene žalbe i da izvrši predikcije o intenzitetu žalbi u narednom mesecu i odgovoru korisnika na razrešenu žalbu, u realnom vremenu. U svrhu predikcije intenziteta žalbi iskorišćen je model linearne regresije, koji je bez ikakvog podešavanja parametara rezultirao vrednošću R^2 mere od 0.64. Za predikciju odgovora korisnika na razrešenje žalbe, upotrebljen je *XGBoost* model, čije vrednosti parametara su određene kombinacijom *RandomizedSearchCV* i *GridSearchCV* algoritma. Ovaj model evaluiran je metodom unakrsne evaluacije, pri čemu se kao metrike evaluacije uzimaju odziv i tačnost. Vrednost mere odziva

prelazi 85%, dok je tačnost iznad 75%. Dodatna prednost *XGBoost* modela jeste ta da je u stanju da prikaže udeo svakog pojedinačnog atributa u predikciji, što je, pored mogućnosti oba modela da pružaju predikcije u realnom vremenu, od izuzetnog značaja za krajnjeg korisnika (kompaniju).

Problem predikcije intenziteta žalbi može se posmatrati kao problem analize vremenskih serija, te bi upotreba modela poput *ARIMA* modela, mogla pružiti preciznije predikcije i na taj način poboljšati ovaj deo sistema. Još jedan od mogućih pravaca unapređenja sistema jeste drugačiji odabir klasifikatora prilikom predikcije korisnikovog odgovora na razrešenu žalbu. Nedavno objavljeni - *BERT* model [15] pokazuje *state-of-the-art* rezultate na *NLP* (*Natural Language Processing*) baziranim problemima.

5. LITERATURA

- [1]<https://www.consumerfinance.gov/data-research/consumer-complaints/> [pristupljeno 16.7.2019.]
- [2]<https://hadoop.apache.org/> [pristupljeno 15.7.2019.]
- [3]<https://spark.apache.org/> [pristupljeno 15.7.2019.]
- [4]<https://restfulapi.net/> [pristupljeno 16.7.2019.]
- [5]<https://angular.io/> [pristupljeno 16.7.2019.]
- [6]<https://dev.socrata.com/foundry/data.consumerfinance.gov/s6-ew-h6mp> [pristupljeno 16.7.2019.]
- [7]https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html [pristupljeno 16.7.2019.]
- [8]<https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html> [pristupljeno 15.7.2019.]
- [9] Clayton J. Hutto / Eric Gilbert, *Vader: A parsimonious rule-based model for sentiment analysis of social media text*, In: Eighth international AAAI conference on weblogs and social media, 2014.
- [10]https://imbalanced-learn.readthedocs.io/en/stable/generated/imblearn.over_sampling.SMOTE.html [pristupljeno 16.7.2019.]
- [11]<https://xgboost.readthedocs.io/en/latest/> [pristupljeno 16.7.2019.]
- [12]https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html [pristupljeno 16.7.2019.]
- [13]https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html [pristupljeno 16.7.2019.]
- [14] Jacob Devlin et al, *Bert: Pre-training of deep bidirectional transformers for language understanding*, arXiv preprint arXiv:1810.04805, 2018.

Kratka biografija:



Milica Nikolić rođena je u Novom Sadu 1994. godine. Osnovne akademske studije iz oblasti *Računarstvo i automatika*, završila je na Fakultetu tehničkih nauka, 2017. godine. kada upisuje i master akademske studije iz iste oblasti. Master rad odbranila je 2019 godine.

kontakt: milka94ns@gmail.com

¹ Granica je određena zahtevom kompanije koja koristi razvijeni sistem.

POREĐENJE PERFORMANSI MATRIČNOG I ALGORITMA SUMIRANJA STRUJA ZA PRORAČUN REŽIMA DISTRIBUTIVNE MREŽE SA KRATKIM SPOJEM

MATRIX AND THE CURRENT SUMMATION ALGORITHM PERFORMANCE COMPARISON FOR SHORT-CIRCUIT CALCULATION IN DISTRIBUTION NETWORK

Milica Delić, Luka Strezoski, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu su upoređeni vremenski zahtevi za proračun režima distributivnih mreža s kratkim spojem primenom dva različita algoritma. Prvi je algoritam sumiranja struja i korekcija napona (SSKN) koji se uobičajeno koristi za proračune u distributivnim mrežama, dok je drugi matrični algoritam koji se uobičajeno koristi za proračune u prenosnim mrežama. Upoređena su vremena potrebna za proračun tri velike, realne, distributivne mreže, koje sadrže 1000, 5000 i 10000 čvorova, respektivno. Cilj ovog poređenja je bio da se ispita mogućnost korišćenja znatno robusnijeg i izuzetno tačnog matričnog algoritma, za proračun distributivnih mreža velikih dimenzija, pogođenih kratkim spojem.

Ključne reči: Distributivne mreže, Matrični algoritam, Vreme izvršenja

Abstract – In this paper, the time requirements for the short-circuit calculation of distribution network using two different algorithms are compared. The first is the current summation and voltage correction algorithm (CSVC) that is commonly used for distribution networks, while another is a matrix algorithm commonly used for calculations in transmission networks. The time required for the calculation of three large, real, distribution networks, containing 1000, 5000 and 10000 nodes, respectively, were compared. The aim of this comparison was to examine the possibility of using a much more robust and extremely accurate matrix algorithm for the calculation of large-scale distribution networks affected by short-circuits.

Keywords: Distribution networks, Matrix algorithm, Execution time

1. UVOD

Proračun režima s kratkim spojem može da se koristi za sebe, ali mnogo češće se koriste u okviru velikog broja ostalih proračuna u sistemima za vođenje prenosnih i distributivnih mreža („Energy Management System – EMS“ i „Distribution Management System – DMS“ softverima) [1]. Izbor prekidačke opreme, podešenje i koordinacija relejne zaštite kao i određivanje lokacija kvara samo su neke od funkcija u okviru kojih je neophodno proračunavati struje kvara ili celog režima mreže sa kvarom i po više puta u toku jednog poziva. Kako je zbog praktične primene pojedinih funkcija bitna brzina izvršenja, vreme potrebno za proračun struja kvara trebalo bi svesti na minimum. Poslednje dve decenije, pri proračunu režima distributivnih mreža sa kratkim spojem,

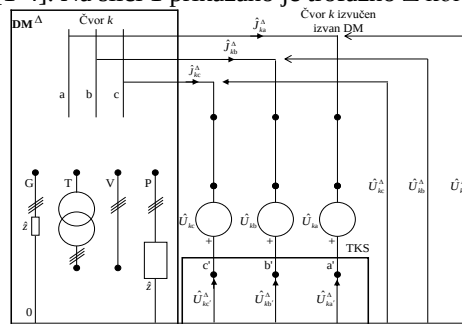
široko je rasprostranjena metoda orijentisana prema granama, poznata kao algoritam sumiranja struja i korekcija napona (SSKN) [2-4]. Zbog specifične topologije distributivne mreže ovaj postupak se pokazao kao vrlo efikasan i brz.

Matrični pristup proračunu režima sa kvarom koji je orijentisan prema čvorovima zbog velikog broja čvorova u distributivnoj mreži u praksi nije našao primenu. [5-6].

U ovom radu se istražuju mogućnosti primene matričnog algoritma, karakterističnog za prenosne mreže, sa svim svojim prednostima u odnosu na algoritme orijentisane prema granama, za proračun režima distributivne mreže sa kvarom, u pogledu vremenskih zahteva. Matrični algoritam može bez problema da uvaži prekid nulte komponente, za razliku od algoritama koji su orijentisani prema granama (npr. algoritam SSKN), pa se zbog toga pri korišćenju algoritama orijentisanih prema granama uvode određene aproksimacije koje konačno dovode do rezultata koji nisu savršeno tačni. Ovaj problem bi se potpuno izbegao ukoliko bi se za proračune distributivnih mreža s kratkim spojem koristio matrični algoritam. Međutim, poznato je da su vremenski zahtevi za izvršenje matričnog algoritma veći, zbog potrebe za inverzijom matrice velikih dimenzija i da sa porastom broja čvorova mreže ti zahtevi rastu. U ovom radu se kvantifikuje koliko je vremena potrebno žrtvovati da bi se primenio jedan izuzetno dobar algoritam (matrični) koji nema problema sa uvažavanjem prekida nulte komponente u distributivnim mrežama velikih dimenzija.

2. MATEMATIČKI MODEL Δ KOLA

Nezavisno od tipa algoritma koji se primenjuje za proračun kratkih spojeva u distributivnim mrežama, vrši se uobičajena dekompozicija režima sa kratkim spojem na režim distributivne mreže pre kratkog spoja i režim tzv. fiktivnog Δ kola, koji je u opštem slučaju trofazan nesimetričan sa nesimetrijom istog karaktera kao što je i nesimetrija režima distributivne mreže sa kratkim spojem [1-4]. Δ kolo je pasivno svuda osim na mestu kratkog spoja [1-4]. Na slici 1 prikazano je trofazno Δ kolo [1].



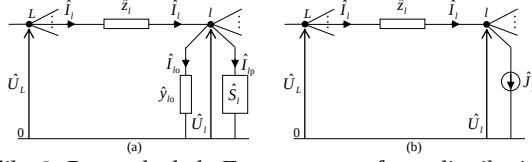
Slika 1. Trofazno Δ kolo

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio doc. dr Luka Strezoski.

2.1. Matematički model izveden direktnom primenom Kirchhoff-ovih zakona

Pogonska kola redosleda s ($s = \text{direktni, inverzni, nulti} - d, i, o$) trofazne Δ mreže razmatrane radialne distributivne mreže lestvičasta su kola. Lestvičasta kola se prikazuju koristeći se Γ segmentima (slika 2) i memorišu koristeći se principom slojeva [7]. Njihovi modeli mogu se izvesti direktnom primenom Kirchhoff-ovih zakona.



Slika 2. Pogonska kola Γ_l segmenta trofazne distributivne mreže (a) i njihova ekvivalentna kola (b)

Dakle, opisani (potpuni) matematički model trofaznog Δ kola sastoji se od sledećih $6n+6$ jednačina [1].

Opšti model Δ kola:

$$\hat{I}_l^{\Delta s} = \hat{J}_l^{\Delta s} + \hat{Y}_{l0}^s \hat{U}_l^{\Delta s} + \sum_{j \in \alpha_l} \hat{I}_j^{\Delta s}, \quad l = n \dots 3, 2 \quad (1)$$

$$\hat{U}_l^{\Delta s} = \hat{U}_L^{\Delta s} - \hat{z}_l^s \hat{I}_l^{\Delta s}, \quad l = 2, 3, \dots, n \quad (2)$$

pri čemu je sa α_l označen skup indeksa svih Γ segmenata kojima je segment Γ_l prethodnik, sa $\hat{I}_l^{\Delta s}$ struja po grani l , sa $\hat{J}_l^{\Delta s}$ injektirana struja u čvor l , sa $\hat{Y}_{l0}^s$ ukupna otočna admitansa u čvoru l , sa $\hat{U}_l^{\Delta s}$ napon čvora l , sa $\hat{I}_j^{\Delta s}$ struja po grani j , sa $\hat{U}_L^{\Delta s}$ napon čvora L , sa $\hat{z}_l^s$ redna impedansa grane l , dok je sa n označen broj čvorova. Oznaka s se odnosi na redosled (direktni, inverzni, nulti – d, i, o).

Zatim relacije kojima se modeluje ostatak Δ kola:

$$F_1(\hat{U}_{ka}^{\Delta}, \hat{U}_{kb}^{\Delta}, \hat{U}_{kc}^{\Delta}, \hat{J}_{ka}^{\Delta}, \hat{J}_{kb}^{\Delta}, \hat{J}_{kc}^{\Delta}) = 0 \quad (3)$$

$$F_2(\hat{U}_{ka}^{\Delta}, \hat{U}_{kb}^{\Delta}, \hat{U}_{kc}^{\Delta}, \hat{J}_{ka}^{\Delta}, \hat{J}_{kb}^{\Delta}, \hat{J}_{kc}^{\Delta}) = 0 \quad (4)$$

$$F_3(\hat{U}_{ka}^{\Delta}, \hat{U}_{kb}^{\Delta}, \hat{U}_{kc}^{\Delta}, \hat{J}_{ka}^{\Delta}, \hat{J}_{kb}^{\Delta}, \hat{J}_{kc}^{\Delta}) = 0 \quad (5)$$

$$\hat{U}_{ka}^{\Delta} - \hat{U}_{kb}^{\Delta} = \hat{U}_{ka}^{\Delta} \quad (6)$$

$$\hat{U}_{kb}^{\Delta} - \hat{U}_{kc}^{\Delta} = \hat{U}_{kb}^{\Delta} \quad (7)$$

$$\hat{U}_{kc}^{\Delta} - \hat{U}_{ka}^{\Delta} = \hat{U}_{kc}^{\Delta} \quad (8)$$

$$\begin{bmatrix} \hat{U}_{ka}^{\Delta} \\ \hat{U}_{kb}^{\Delta} \\ \hat{U}_{kc}^{\Delta} \end{bmatrix} = \hat{A} \begin{bmatrix} \hat{U}_{ka}^{\Delta} \\ \hat{U}_{kb}^{\Delta} \\ \hat{U}_{kc}^{\Delta} \end{bmatrix}, \quad (9) \quad \begin{bmatrix} \hat{J}_{ka}^{\Delta} \\ \hat{J}_{kb}^{\Delta} \\ \hat{J}_{kc}^{\Delta} \end{bmatrix} = \hat{A} \begin{bmatrix} \hat{J}_{ka}^{\Delta} \\ \hat{J}_{kb}^{\Delta} \\ \hat{J}_{kc}^{\Delta} \end{bmatrix}. \quad (10)$$

pri čemu je sa $\hat{A}$ označena matrica transformacije iz faznog u domen simetričnih komponenti, sa $\hat{U}_{ka}^{\Delta}, \hat{U}_{kb}^{\Delta}, \hat{U}_{kc}^{\Delta}$ naponi čvora k u tačkama $a', b' i c'$, sa $\hat{J}_{ka}^{\Delta}, \hat{J}_{kb}^{\Delta}, \hat{J}_{kc}^{\Delta}$ injektirane struje u čvoru k , sa $\hat{U}_{ka}^{\Delta}, \hat{U}_{kb}^{\Delta}$ i $\hat{U}_{kc}^{\Delta}$ naponi idealnih naponskih izvora insertovanih u ansambl kratkog spoja, sa $\hat{U}_{ka}^{\Delta}, \hat{U}_{kb}^{\Delta}, \hat{U}_{kc}^{\Delta}$ naponi čvora k u tačkama $a, b i c$, dok su $\hat{U}_{ka}^{\Delta}, \hat{U}_{kb}^{\Delta}, \hat{U}_{kc}^{\Delta}$ i $\hat{J}_{ka}^{\Delta}, \hat{J}_{kb}^{\Delta}, \hat{J}_{kc}^{\Delta}$ simetrične komponente napona i struja $\hat{J}_{ka}^{\Delta}, \hat{J}_{kb}^{\Delta}, \hat{J}_{kc}^{\Delta}$, (slika 1).

2.2. Matematički model napisan u saglasnosti sa metodom nezavisnih napona

Pogonska kola redosleda s ($s=d, i, o$) trofazne Δ mreže razmatrane radialne distributivne mreže lestvičasta su kola.

Lestvičasta kola se prikazuju koristeći se Γ segmentima (slika 2) i memorišu koristeći se principom slojeva. Njihovi modeli mogu se izvesti primenom metode nezavisnih napona.

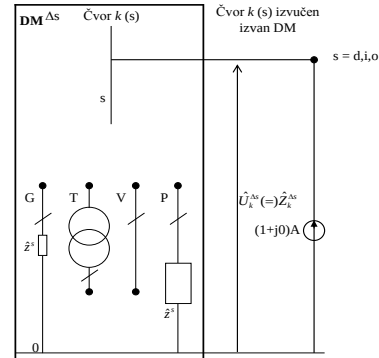
Opšti model Δ kola:

$$\hat{J}_{n-1,1}^{\Delta s} = \hat{Y}_{n-1,n-1}^s \hat{U}_{n-1,1}^{\Delta s}, \quad s = d, i, o \quad (11)$$

pri čemu je sa $\hat{J}_{n-1,1}^{\Delta s}$ označen vektor injektiranih struja u čvorove, sa $\hat{Y}_{n-1,n-1}^s$ matrica admitansi sistema, sa $\hat{U}_{n-1,1}^{\Delta s}$ označeni naponi čvorova, sa n broj čvorova. Oznaka s se odnosi na redosled (direktni, inverzni, nulti – d, i, o). Pored ovih relacija, potpun matematički model čine i relacije kojima se modeluje ostatak Δ kola, a koje su označene sa (3,4,5,6,7,8,9,10).

3 REŠENJE MATEMATIČKOG MODELA Δ KOLA

Nezavisno od tipa algoritma koji se primenjuje za proračun kratkih spojeva u distributivnim mrežama, prvo je neophodno izračunati ekvivalente impedanse gledano odočeno sa mesta kvara za direktni, inverzni i nulti režim. Pogonsko kolo trofaznog Δ kola, koje je prikazano na slici 3, pasivno je svuda osim na mestu kratkog spoja (čvor k). Tako, elektromotorne sile njihovih Thévenin-ovih ekvivalenata jednake su nuli, za svaku vremensku sekvencu. Thévenin-ova impedansa pogonskog kola redosleda $s, s=d, i, o$, za bilo koju vremensku sekvencu, brojno je određena naponom čvora k pogonskog kola redosleda s , izabrane vremenske sekvence, kada se između čvora nultog potencijala i čvora k u tom pogonskom kolu, priključi idealan strujni generator jedinične struje – slika 3 [1].



Slika 3. Pogonsko kolo redosleda s trofazne Δ mreže, koje je pripremljeno za izvođenje Thévenin-ovih impedansi

Vrednost ekvivalentne impedanse dobijene na opisan način jednaka je sa vrednošću impedanse sa k -tog dijagonalnog mesta iz matrice impedansi koja se kod matrice algoritma dobija inverzijom matrice admitansi [10].

3.1 Rešenje matematičkog modela izvedenog direktnom primenom Kirchhoff-ovih zakona

S obzirom da je režim distributivne mreže pre kratkog spoja poznat, poznat je i napon $\hat{U}_{ka}^{\Delta}$ faze a čvora s kratkim spojem k , pre kratkog spoja. S obzirom na to da je struja na mestu kvara funkcija napona pre kratkog spoja i ekvivalentnih impedansi (20), kada se izračunaju Thévenin-ove impedanse pogonskih kola trofaznog Δ kola ($\hat{Z}_k^d, \hat{Z}_k^i, \hat{Z}_k^o$; za impedansu direktnog redosleda $\hat{Z}_k^d$ računaju se tri vrednosti, po jedna za svaku od tri vremenske sekvence), onda se mogu izračunati ključne nepoznate veličine – simetrične komponente struja ansambla kratkog spoja u Δ kolu: $\hat{J}_k^{\Delta d}, \hat{J}_k^{\Delta i}, \hat{J}_k^{\Delta o}$, za bilo koju vremensku sekvencu.

$$\hat{J}_k^{\Delta s} = f(\hat{U}_{ka}^{\Delta}, \hat{Z}_k^d, \hat{Z}_k^i, \hat{Z}_k^o), \quad s=d, i, o \quad (12)$$

Kada se te tri struje poznaju, tada u 6n-6 jednačine (1) i (2) ima isto toliko nepoznatih veličina:

- 1) $\hat{I}_l^{As}, l = 2, 3, \dots, n, s = d, i, o$; ima ih 3n-3;
- 2) $\hat{U}_l^{As}, l = 2, 3, \dots, n, s = d, i, o$; ima ih 3n-3.

Te algebarske, linearne, simultane i kompleksne jednačine treba rešiti, pre svega, po 3n veličina stanja – simetrične komponente napona Δ kola: $\hat{U}_l^{As}, l = 2, 3, \dots, n, s = d, i, o$.

One se mogu rešiti različitim metodima. Za njihovo rešenje se koristi Gauss/Seidel-ov iterativni metod, odnosno postupak SSKN. Jednačine (1) i (2) su pripremljene za direktnu primenu tog metoda:

$$(\hat{I}_l^{As})^{h+1} = \hat{J}_l^{As} + \hat{Y}_{ls}^{As} (\hat{U}_l^{As})^h + \sum_{j \in \alpha_l} (\hat{I}_j^{As})^{h+1}, l = n \dots 3, 2 \quad s = d, i, o, \quad (13)$$

$$(\hat{U}_l^{As})^{h+1} = (\hat{U}_l^{As})^{h+1} - \hat{Z}_l^{As} (\hat{I}_l^{As})^{h+1}, l = 2, 3, \dots, n, s = d, i, o, \quad (14)$$

pri čemu su struje $\hat{I}_l^{As}$ poznate:

$$\hat{J}_l^{As} = \begin{cases} 0, & \text{za } l \neq k \\ \hat{J}_k^{As} \neq 0, & \text{za } l = k \end{cases}, \quad s = d, i, o \quad (15)$$

Sa h je označen redni broj iteracije Gauss/Seidel-ovog iterativnog metoda ($h=1, 2, 3 \dots$). Simetrične komponente napona $\hat{U}_l^{As}, s = d, i, o$, za $L=1$ (prvi čvor segmenta korena – Thévenin-ovog ekvivalenta nadređene mreže koji je priključen u korenu – između čvorova 1 i 2), jednake su nuli: $\hat{U}_1^{As} = 0, s = d, i, o$. Tako, one se ne menjaju od iteracije do iteracije: $(\hat{U}_1^{As})^{h+1} = 0, \forall h$.

Za početne vrednosti simetričnih komponenti napona trofaznih čvorova Δ kola mogu se usvojiti nulte vrednosti:

$$(\hat{U}_l^{As})^1 = 0, \quad l = 2, 3, \dots, n, \quad s = d, i, o \quad (16)$$

Iterativna procedura se zaustavlja kada se zadovolje sledeći kriterijumi konvergencije:

$$\left| (\hat{I}_l^{As})^{h+1} - (\hat{I}_l^{As})^h \right| \leq \varepsilon_i \wedge \left| (\hat{U}_l^{As})^{h+1} - (\hat{U}_l^{As})^h \right| \leq \varepsilon_u \wedge \left| (\hat{U}_l^{As})^{h+1} - (\hat{U}_l^{As})^h \right| \leq \varepsilon_u, \quad (17)$$

pri čemu su $\varepsilon_i, \varepsilon_u$ i ε_u kriterijumi konvergencije (po želji izabrani pozitivni brojevi). Najčešće je dovoljno proveravati samo poslednja dva od tri kriterijuma (17) [1].

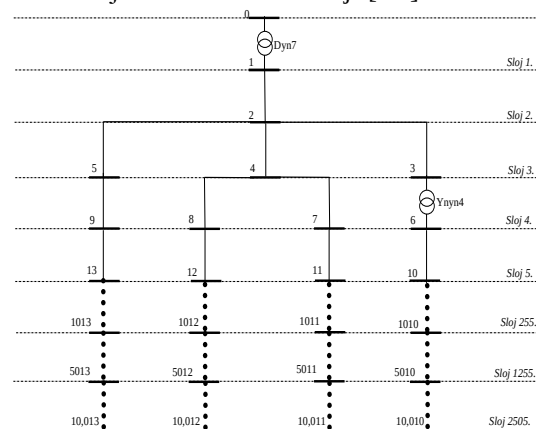
3.2 Rešenje matematičkog modela napisanog u saglasnosti sa metodom nezavisnih napona

S obzirom da je režim distributivne mreže pre kratkog spoja poznat, poznat je i napon $\hat{U}_{ka}$ faze a čvora s kratkim spojem k , pre kratkog spoja. Kada se izračunaju Thévenin-ove impedanse pogonskih kola trofaznog Δ kola ($\hat{Z}_k^d, \hat{Z}_k^i, \hat{Z}_k^o$; za impedansu direktnog redosleda $\hat{Z}_k^d$ računaju se tri vrednosti, po jedna za svaku od tri vremenske sekvence), onda se mogu izračunati ključne nepoznate veličine – simetrične komponente struja ansambla kratkog spoja u Δ kolu: $\hat{J}_k^{Ad}, \hat{J}_k^{Ai}, \hat{J}_k^{Ao}$, za bilo koju vremensku sekvencu. Kada se te tri struje poznaju, tada se poznaju vektori s levih strana tri skupa od po n jednačina – opšteg modela pogonskih kola trofaznog Δ kola u admitantnoj formi. Svaki od ta tri skupa od po n jednačina može se rešiti po n nepoznatih odgovarajućih simetričnih komponenti napona n trofaznih čvorova Δ kola, nezavisno od druga dva skupa po n jednačina. Tim rešenjima su obuhvaćene i simetrične komponente napona samog čvora k sa kratkim spojem. Dakle, kada se raspolaze sa vrednostima te tri struje problem rešavanja sistema od 3n+12 spregnutih jednačina i

isto toliko nepoznatih veličina se svodi na rešavanje tri raspregnuta sistema, svaki od po n linearnih jednačina sa n nepoznatih veličina. Oni su napisani u formi nad kojom se za njihovo rešenje može direktno primeniti Gausova redukcija. Rešavanje tri raspregnuta sistema jednačina je znatno jednostavnije od rešavanja 3n+12 spregnutih jednačina. Korišćenjem opisanog postupka, mogu se izračunati simetrične komponente napona u svim čvorovima Δ kola. Tih 3n napona – rešenje Δ kola – predstavljaju stanje naizmenične komponente režima trofaznog Δ kola u domenu simetričnih komponenti za izabranu vremensku sekvencu. Iz njih se prema definiciji stanja može izračunati celokupni režim Δ kola. Zatim se taj režim može transformisati u fazni domen.

4 NUMERIČKA VERIFIKACIJA PRORAČUNA

Trofazna distributivna mreža nad kojom je vršena numerička verifikacija proračuna prikazana je na slici 4. Mreža se napaja preko prenosne mreže koja je modelovana mrežom beskonačne snage. Sastoji se od vodova i transformatora. Svi parametri su dati u domenu simetričnih komponenti. Izvršena je normalizacija kao i Θ -transformacija [8-9].



Slika 4. Distributivna test mreža

Mreža se sastoji od 14 čvorova i 13 rednih grana. Svi vodovi su jednakih dužina i parametara. Dužina svakog voda je 0.0914 [km], a redni i otočni parametri su dati u tabelama 1 i 2.

Tabela 1 – Podužni redni parametri vodova

Z (R + jX) [Ω/km]			
faza	b	c	a
a	0.828+j0.848	0.129+j0.287	0.128+j0.285
b	0.129+j0.287	0.828+j0.848	0.129+j0.287
c	0.128+j0.285	0.129+j0.287	0.828+j0.848

Tabela 2 – Podužni otočni parametri vodova

Y (G + jB) [μS/km]			
faza	a	b	c
a	0.000+j2.916	0.000-j0.563	0.000-j0.559
b	0.000-j0.563	0.000+j2.916	0.000-j0.563
c	0.000-j0.559	0.000-j0.563	0.000+j2.916

Podaci o transformatorima su dati u tabeli 3.

Tabela 3 – Podaci o transformatorima

Tr	Sprega	m _k [kV/kV]	Sn [kVA]	R [%]	X [%]
1	Dyn	115/4.16	5000	1.0	8.0
2	Ynyn	4.16/0.48	500	1.1	2.0

Simulacije kratkih spojeva vršene su u programu koji je napisan i pokretan na računaru koji ima 4GB RAM memorije i procesor brzine 2.1 GHz. Za simuliranje jednofaznog kratkog spoja izabrana je faza a. Kratki spojevi su simulirani u svim čvorovima mreže primenom oba algoritma. Vrednosti struja kvara dobijene jednim i drugim algoritmom su jednake i date su u tabeli 4.

Tabela 4 – Struje Δ kola na mestu kvara pri 1FKS

Čvor	Faza					
	a		b		c	
	I[A]	ugao[°]	I[A]	ugao[°]	I[A]	ugao[°]
1	8600.16	-82.87	0.0	0.0	0.0	0.0
2	6479.14	-72.71	0.0	0.0	0.0	0.0
3	5121.46	-66.70	0.0	0.0	0.0	0.0
4	5120.94	-66.69	0.0	0.0	0.0	0.0
5	5121.29	-66.70	0.0	0.0	0.0	0.0
6	16547.33	-63.23	0.0	0.0	0.0	0.0
7	4208.83	-62.79	0.0	0.0	0.0	0.0
8	4208.83	-62.79	0.0	0.0	0.0	0.0
9	4209.07	-62.79	0.0	0.0	0.0	0.0
10	15296.47	-61.86	0.0	0.0	0.0	0.0
11	3562.84	-60.08	0.0	0.0	0.0	0.0
12	3562.84	-60.08	0.0	0.0	0.0	0.0
13	3563.02	-60.08	0.0	0.0	0.0	0.0

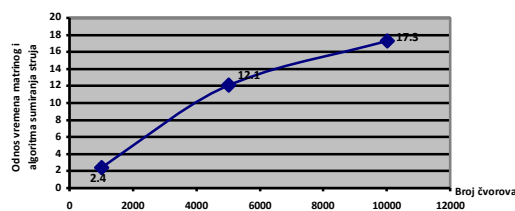
Na svaku od 4 završne grane iz poslednjeg sloja iz gore opisane mreže od 14 čvorova dodato je još 250 grana jednakih parametara i na taj način formirana mreža od 1014 čvorova. Na isti način mreža je proširena na 5014, a zatim i na 10014 čvorova.

Na svakoj od ovih mreža je vršena simulacija kratkih spojeva i merena su vremena potrebna za izračunavanje algoritmom SSKN, a zatim matričnih algoritmom. Vremena su data u tabeli 5.

Tabela 5 – Vremena trajanja proračuna

Broj čvorova	Vreme [s]	
	Algoritam SSKN	Matrični algoritam
1014	0.9375	2.2500
5014	20.8281	252.3281
10014	107.3750	1854.8440

Na osnovu rezultata izloženih u ovom radu, vidi se da je vreme potrebno za proračun kratkih spojeva matričnim algoritmom veće od vremena potrebnog za proračun algoritmom SSKN. Ta razlika nije mnogo izražena na mreži od 1014 čvorova. Za matrični algoritam je potrebno $2.25[s]/0.9375[s] = 2.4$ puta više vremena. Kako se povećava broj čvorova tako raste i razlika u potrebnom vremenu. Za proračun matričnim algoritmom na mreži od 5014 čvorova potrebno je $252.3281[s]/20.8281[s] = 12.1$ puta više vremena, dok je na mreži od 10014 čvorova potrebno $1854.844[s]/107.375[s] = 17.3$ puta više vremena. Odnos vremena potrebnih za proračun matričnim i algoritmom SSKN u zavisnosti od broja čvorova grfički je prikazan na slici 5.



Slika 5. Odnos vremena potrebnog za proračun u zavisnosti od broja čvorova

Ako se uzme u obzir skromna konfiguracija računara sa kojim je ovaj proračun vršen (4GB RAM memorije i 2.1 GHz procesor) i dobijeni rezultati, može se zaključiti da bi se ovaj odnos prilično smanjio kada bi se proračun vršio na današnjim industrijskim računarima sa znatno moćnijom konfiguracijom, na kojima se i vrše praktični DMS proračuni. Na osnovu ovih rezultata, može se pretpostaviti da bi se vremena dodatno približila, ali svakako i značajno smanjila, te vremenski zahtevi vrlo verovatno ne bi bili ograničavajući faktor za primenu

matričnog algoritma za proračun kratkih spojeva u velikim distributivnim mrežama. Da bi se ove pretpostavke i potvrdile sa sigurnošću, potrebno je odraditi dodatno istraživanje na ovu temu. Potvrda navedenih pretpostavki jeste jedan od pravaca za dalja istraživanja autora.

5 ZAKLJUČAK

Cilj ovog poređenja matričnog i algoritma SSKN bio je da se ispita mogućnost korišćenja znatno robusnijeg i izuzetno tačnog matričnog algoritma, za proračun distributivnih mreža velikih dimenzija, pogođenih kratkim spojem. Korišćenjem matričnog algoritma izbegle bi se neophodne aproksimacije koje se uvode korišćenjem SSKN algoritma, pa bi se samim tim i povećala tačnost rezultata. Poređenjem vremenskih zahteva za proračun kratkih spojeva na distributivnim mrežama koje sadrže 1014, 5014 i 10014 čvorova, respektivno, prvo algoritmom SSKN, zatim matričnim algoritmom, na računaru skromne konfiguracije (4GB RAM memorije i 2.1 GHz procesor), ipak se pokazuje da je SSKN algoritam brži i da razlika u brzini raste sa porastom broja čvorova, međutim i pored toga, rezultati otvaraju mogućnost primene matričnog metoda u industrijskim softverima. Kada se uzmu u obzir znatna poboljšanja računarskih performansi u poslednjoj deceniji, može se zaključiti da bi se pri proračunima matričnim metodom vršenim u industrijskim softverima, na računarima sa znatno moćnijom konfiguracijom, vremenski zahtevi znatno smanjili i približili vremenskim zahtevima za proračun algoritmom SSKN. Ovim se otvara mogućnost primene matričnog algoritma za proračun režima realnih distributivnih mreža pogođenih kvarom u industrijskim softverima, sa aspekta vremenskih zahteva.

6 LITERATURA

- [1] L. V. Strezoski and M. D. Prica, "Real-time short-circuit analysis of active distribution systems," *2016 IEEE Power and Energy Conference at Illinois (PECI)*, Urbana, IL, 2016, pp. 1-8.
- [2] X.Zhang, F.Soudi, D.Shirmohammadi, C.Cheng: A Distribution Short Circuit Analysis Approach Using Hybrid Compensation Method, *IEEE Transactions on Power Systems*, Vol. 10, No. 4, 1995., pp. 2053-2059.
- [3] A.Tan, W.H.Liu, D.Shirmohammadi: Transformer and Load Modeling in Short Circuit Analysis for Distribution Systems, *IEEE Transactions on Power Systems*, Vol. 12, No. 3, 1997., pp. 1315-1322.
- [4] D.Rajicic, R.Taleski: Two Novel Methods for Radial and Weakly Meshed Network Analysis, *Electric Power System Research*, 48, 1998., pp. 79-87.
- [5] V.Strezoski, D.Bekut: A Canonical Model for the Study of Faults in Power Systems, *IEEE Transactions on Power Systems*, Vol. 6, No. 4, 1991., pp. 1493-1499.
- [6] V.C.Strezoski, G.S.Švenda, D.D.Bekut: Extension of the Canonical Model to Grounding Parts of Power Systems Under Fault Conditions, *Electrical Power & Energy Systems*, No 25, 2003, pp 567-575.
- [7] P. Vidovic, Proračuni tokova snaga neuravnoteženih distributivnih mreža, doktorska disertacija, Fakultet Tehničkih Nauka, Univerzitet u Novom Sadu, Novi Sad, 2015.
- [8] V.Strezoski: A New Scaling Concept in Power System Analysis, *IEE Proc.-Gener. Transm. Distrib.*, Vol. 143, No. 5, 1996., pp. 399-406.
- [9] V. C. Strezoski: Advanced symmetrical components method, *IET Proc.-Gener. Transm. Distrib.*, Vol. 5, Issue 8, ISSN 1751-8687, August 2011, pp. 833-841.
- [10] V.C.Strezoski: Osnovni proračuni elektroenergetskih sistema, Fakultet Tehničkih Nauka, Novi Sad, 2017.

**MODEL I SIMULACIJE VETROGENERATORA U ATP/EMTP-U I
MATLAB/SIMULINK-U ZA SLUČAJ ATMOSFERSKOG PRAŽNJENJA****MODEL AND SIMULATIONS OF WIND TURBINE IN ATP/EMTP AND
MATLAB/SIMULINK IN CASE OF ATMOSPHERIC DISCHARGE**Milana Komisar¹, Michael Rock², Vladimir Katić³;^{1,3}Fakultet tehničkih nauka, Novi Sad, Srbija; ²Tehnički Univerzitet Ilmenau, Ilmenau, Nemačka**Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – U radu je razmatran uticaj atmosferskog pražnjenja na vetrogenerator. Modelovan je sistem vetrogeneratora u dva različita softverska alata, u ATP/EMTP-u i Matlab/Simulink-u. Posmatrani su i upoređeni rezultati napona i struje u sistemu prilikom udarne struje.

Ključne reči: Atmosfersko pražnjenje, Vetrogenerator, ATP/EMTP, Matlab/Simulink

Abstract - In this paper, the impact of atmospheric discharge on a wind turbine is presented. Wind turbine is modelled using two different software tools ATP/EMTP and Matlab/Simulink. The current and voltage distribution during lightning impulse current strikes are observed and results are compared.

Keywords: Atmospheric Discharge, Wind Turbine, ATP/EMTP, Matlab/Simulink

1. UVOD

U svetu, a i kod nas, sve je veća zastupljenost distribuiranih elektroenergetskih izvora baziranih na obnovljivim izvorima energije. U tu grupu ubrajaju se i vetrogeneratori, čiji kapaciteti se ubrzano povećavaju, tako da danas predstavljaju najznačajnije ovakve elektrane. Kao objekti specifične konstrukcije, sa veoma visokim stubom (tornjem) i dugačkim oštricama lopatica vetroturbine (rotora), kao i zbog njihovih karakterističnih lokacija, vetrogeneratori su pogodna meta udara atmosferskih pražnjenja.

U radu su razmatrani efekti, koji se javljaju pri udarnim strujama na elementima vetrogeneratorskog sistema i pojave prenapona na određenim elementima, kroz modelovanje i simulacije korišćenjem dva poznata softverska alata ATP/EMTP i Matlab/Simulink. Dobijeni rezultati su upoređeni i komentarisani.

NAPOMENA:

Ovaj rad proistekao je iz master rada, koji je rađen na Tehničkom Univerzitetu u Ilmenau (Nemačka) u sklopu studentske razmene (Projekat Erasmus+) i čiji supervizor je bio prof. dr Michael Rock, a odbranjen na FTN u Novom Sadu, pod mentorstvom prof. dr Vladimira A. Katića.

2. ATMOSFERSKO PRAŽNJENJE

Atmosfersko pražnjenje je vremenski fenomen povezan sa masivnim električnim pražnjenjem u nižim slojevima atmosfere, pri površini zemlje koji je praćen zvučnim i svetlosnim efektom, tj. grmljavinom i bljeskom (munjom). Između 2000 and 3000 atmosferskih pražnjenja je konstantno aktivno širom sveta, oko 30 do 120 po sekundi. Pojavljuju se udarne struje amplitude preko 200 kA, koje mogu prouzrokovati značajne posledice na ljude, objekte i tehničku opremu [1].

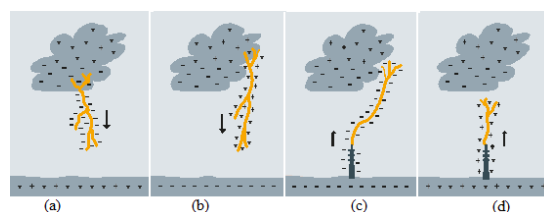
Atmosfersko pražnjenje je suštinski električni proboj vazduha. Javlja se kada jačina električnog polja između oblaka i zemlje premaši električnu čvrstoću vazduha, tj. 3 MV/m [2].

2.1. Vrste atmosferskog pražnjenja

Postoje tri osnovne vrste atmosferskog pražnjenja, u skladu sa tim šta se nalazi na krajevima kanala koji se manifestuje tom prilikom. Tako postoje atmosferska pražnjenja, koja se javljaju:

- 1) unutar oblaka,
- 2) između dva različita oblaka i
- 3) između oblaka i zemlje.

Dalja podela atmosferskog pražnjenja između oblaka i zemlje jeste prema smeru inicirajućeg naelektrisanja i prema vrsti naelektrisanja koja se sprovodi od oblaka ka zemlji. Mogu se pojaviti lideri na više (od zemlje ka oblaku) i na niže (od oblaka ka zemlji), i oba mogu biti pozitivne ili negativne prirode. Na slici 1 predstavljeni su tipovi atmosferskog pražnjenja između oblaka i zemlje na osnovu smera i naelektrisanja.



Slika 1. Tipovi atmosferskog pražnjenja između oblaka i zemlje na osnovu smera i naelektrisanja

2.2. Tipičan oblik strujnog talasa prouzrokovanog atmosferskim pražnjenjem

U praksi se najčešće koristi Heidlerov model udarne struje [3]. Ovaj model je značajan jer najverodostojnije opisuje

oblik talasa koji se javlja pri atmosferskom pražnjenju. Opisuje se sledećom matematičkom jednačinom:

$$i = \frac{i_{max}}{\eta} * \frac{\left(\frac{t}{T}\right)^{10}}{1 + \left(\frac{t}{T}\right)^{10}} * e^{-\frac{t}{\tau}}, \text{ za } t \geq 0 \quad (1)$$

gde je i_{max} – vršna vrednost struje u A; η – korekcionni faktor temene vrednosti struje; t – vreme u s; T – vremenska konstanta čela u s; τ – vremenska konstanta začelja u s.

Tabela 1. Parametri za analitičko predstavljanje udarne struje [4]

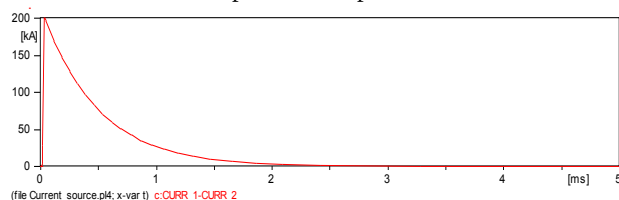
Parametri	Talasni oblik T_1/T_2		
	Prvi pozitivan udar 10/350 μ s	Prvi negativan udar 1/200 μ s	Negativan povratni udar 0,25/100 μ s
η	0,930	0,986	0,993
T	19 μ s	1,82 μ s	0,454 μ s
τ	485 μ s	285 μ s	143 μ s

2.3. Uticaj atmosferskog pražnjenja na vetrogenerator

Vetrogenerator, kao objekat značajne visine (u ovom radu je pretpostavljena visina od 100 m) i na specifično pogodnim pozicijama za iskorišćenje energije vetra je na meti atmosferskog pražnjenja. Oštrice lopatice rotora imaju receptore koji reaguju kao „hvatači“ prilikom udara munje i sprovode strujni impuls kroz provodnike koji su ugrađeni unutar elisa od delova u gondoli koji su dalje povezani sa sistemom uzemljenja, izjednačavanja potencijala i zaštite. Obično su kritične tačke za udar pri vrhu oštrice rotora, iako nisu isključene mogućnosti spoja strujnog kanala sa ostalim delovima sistema. Značajna je energetska vrednost koju nosi udarni impuls i koja može da prouzrokuje ozbiljne posledice termičke i mehaničke prirode.

2.4. Tip atmosferskog pražnjenja korišćen u simulacijama

U ovom radu se koristi prvi visokoenergetski impuls udarne struje, talasnog oblika 10/350 μ s, sa vršnom vrednošću struje do 200 kA (slika 2). Talasni oblik udarne struje pojednostavljen je u svrhu simulacija. Koristi se samo prvi talasni oblik impulsnog udara definisan Heidlerovom formulom i prethodno spomenutim vrednostima.



Slika 2. Talasni oblik impulsne struje groma

3. DELOVI VETROGENERATORA

Osnovni razmatrani delovi vetrogeneratora su: oštrice lopatica rotora, NN generator, konvertor, toranj sa energetskim kablovima unutra, NN-SN transformator, uzemljenje sistema i SN mreža kao spojnica za povezivanje na distributivnu mrežu napona 20 kV. Ovi sklopovi su modelovani i testirani pomoću simulacija najpre zasebno u kolu sa strujnim izvorom, a zatim u kratkom spoju.

4. SOFTVERSKI ALATI ZA MODELOVANJE

Modelovanje je proces u kojem korisnik opisuje dinamički sistem matematičkim jednačinama, a zatim kreira pojednostavljeni prikaz sistema s modelom. Jednačine definišu ponašanje sistemu.

4.1. ATP/EMTP

EMTP (*Electromagnetic Transients Program*) je primarni simulacioni program koji se koristi u elektroenergetici, dok je ATP (*Alternative Transients Program*) verzija ovog softverskog alata.

Zajedno, ATM/EMTP se veoma često koriste kao univerzalni programski sistem za digitalne simulacije prelaznih procesa elektromagnetske i elektromehaničke prirode u EES-u [5].

4.2. Matlab/Simulink

Simulink je okruženje blok dijagrama za simulaciju više domena i dizajniranju modela različitih elemenata sistema. Podržava dizajn, simulaciju, automatsko generisanje koda na nivou sistema i kontinuirano testiranje i verifikaciju ugrađenih sistema.

Simulink nudi grafički editor, prilagodljive blok biblioteke i rešenja za modelovanje i simulaciju dinamičkih sistema [6].

5. MODEL VETROGENERATORA ZA VREME ATMOSFERSKOG PRAŽNJENJA U ATP/EMTP

Na slici 3 prikazan je celokupan model vetrogeneratora koji je izveden u ATP/EMTP softverskom alatu. Integrisani su svi osnovni električni delovi sistema zajedno sa izvorom udarne struje, koju predstavlja Heidlerova funkcija.

Merni instrumenti su postavljeni za posmatranje distribucije struja i napona koji se pojavljuju na značajnim mestima. Od interesa su vrednosti napona i struja na generatoru, transformatoru i energetskim kablovima, prvenstveno radi njihove zaštite.

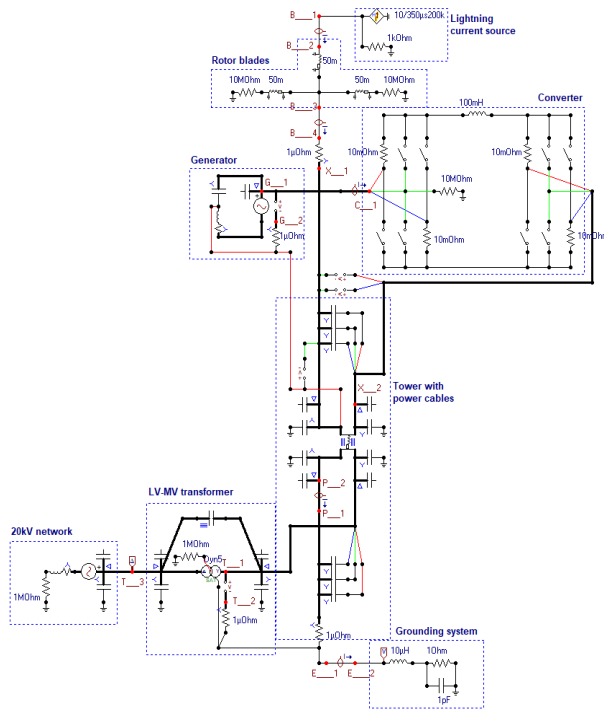
6. MODEL VETROGENERATORA ZA VREME ATMOSFERSKOG PRAŽNJENJA U SIMULINK

Kao u ATP/EMTP i u Matlab/Simulink-u modelovan je ceo sistem vetrogeneratora sa istim električnim delovima i izvorom udarne struje (slika 4).

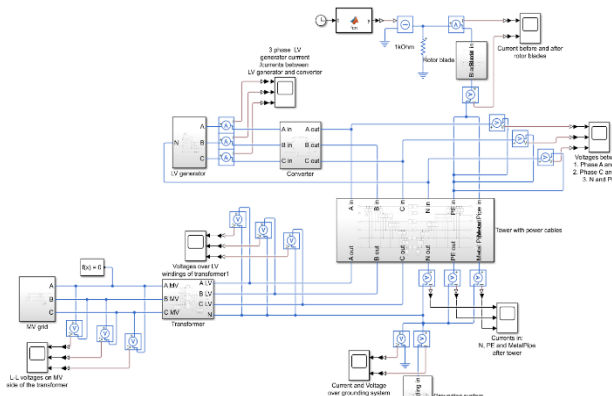
Prilikom modelovanja elemenata, korišćene su iste vrednosti parametara modela i iste šeme. Iz tog razloga su i korišćeni jednostavniji modeli elemenata, da bi se oni što verodostojnije mogli replikovati u oba softverska okruženja.

7. REZULTATI SIMULACIJA

Simulacije u oba program su izvršene za vreme od 5 ms. Linijski elementi odnosno, oštrice rotora i toranj sa kablovima su prvo modelovani kao distribuirani elementi. Rezultati su pokazali da nema vidljivih efekata refleksije putujućeg talasa, jer je vreme čela impulsne struje veće od vremene putujućeg talasa. Na primeru rotorske oštrice vreme putujućeg talasa izračunato je kao:



Slika 3. Model vetrogeneratora tokom udara groma u ATP-u



Slika 4. Model vetrogeneratora tokom udara groma u Simulink-u

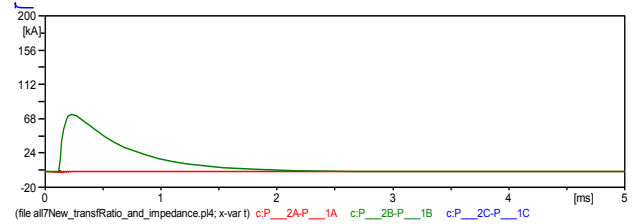
$$T = \frac{l}{c} = \frac{50 \text{ m}}{300 \text{ m}/\mu\text{s}} = 0.167 \mu\text{s} \quad (2)$$

7.1. Rezultati raspodele struja i prenapona

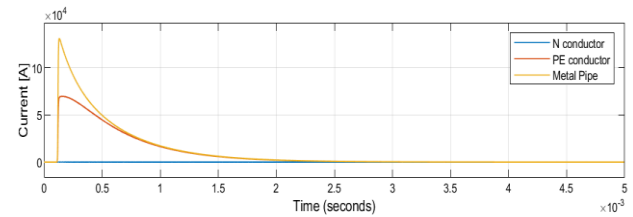
Rezultati koji su dobijeni poređenjem celokupnih modela vetrogeneratora u ATP/EMTP i Simulink-u pokazuju pre svega, iste vrednosti struja i napona u stacionarnom stanju (normalan, operativni rad sistema). Približno ista raspodela struja je rezultat u oba modela.

Međutim, na slikama 5 i 6 je vidljivo da struja merena u podnožju tornja ima blage promene. Uočljiv je pomeraj talasnog oblika struje PE provodnika. Maksimum struje u ovom provodniku u Simulink modelu javlja se nešto ranije (crvena linija na slici 6) nego u ATP-u (zelena linija na slici 5) i sa nešto većom maksimalnom vrednošću. Zbog toga je struja izmerena u metalnom tornju u Simulink modelu manja (žuta linija na slici 6 i plava na slici 5). Po Kirhofovom zakonu, na mestu čvorišta struja atmosferskog pražnjenja ulazi u čvor, a struje u N, PE provodnicima i metalnom tornju izlaze iz čvora i sumarno daju zbir od 200 kA, u oba slučaja.

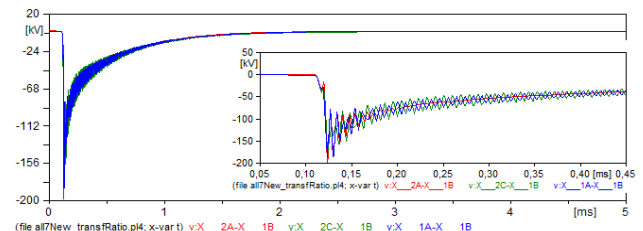
Slike 7 i 8 pokazuju rezultat merenih napona na vrhu tornja između PE provodnika i faza A, C i N, respektivno. Posledično, kako su se pojavile razlike u strujama u ova dva modela, tu su razlike i u naponima. U Simulink-u se pojavljuje znatno veća vršna vrednost izmerenih napona (-1,4 MV), nego u ATP-u (-0,2 MV).



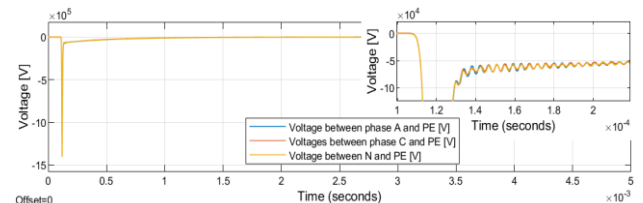
Slika 5. Talasni oblik struja u N, PE provodniku i u metalnoj cevi na dnu tornja u ATP-u



Slika 6. Talasni oblik struja u N, PE provodniku i u metalnoj cevi na dnu tornja u Simulink-u



Slika 7. Napon između PE provodnika i faza A, C i N respektivno u ATP-u



Slika 8. Napon između PE provodnika i faza A, C i N respektivno u Simulink-u

Nastala pojava može da se objasni sledećim izrazom:

$$e_{21}(t) = L_{11} \frac{di_2(t)}{dt} + M_{21} \frac{di_2(t)}{dt} \quad (3)$$

odnosno indukovanjem napona na induktivnosti i međuinaktivnosti provodnog puta.

Ovde je bitno naglasiti da je jedina, značajna razlika u modelima napravljena pri modelovanju tornja kod kablova. U ATP/EMTP toranj sa kablovima je modelovan sa sopstvenim induktivnostima, kapacitivnostima i otpornostima, isto kao što su u modelu uvažene i međusobne induktivnosti i kapacitivnosti. Dok u modelu tornja sa kablovima u Simulinku nisu uvažene međusobne induktivnosti. Razlog tome je nepostojanje već gotovog bloka koji bi uvažio međusobnu induktivnu spregu 6 linijskih provodnika, dok je u EMTP/ATP takav element za modelovanje dostupan.

Sumarno, osnovne razlike u rezultatima ATP i Simulink modela su u opsegu oscilacija i vršnim vrednostima koje se pojavljuju u vreme prednjeg dela (začelja) impulsnog strujnog uzvora (uvećani detalji na slikama 7 i 8).

7.2. Ograničenja i razlike između ATP i Simulink modela

Osnovna razlika između modela u ATP/EMTP i Matlab/-Simulink softverskim alatima jeste izostavljanje induktivne sprege u modelu tornja sa energetske kablovima. Nakon testiranja zasebnih modela tornja sa energetske kablovima u ova dva, različita softvera, dobijeni rezultati nisu sa značajnim ili velikim razlikama.

Međutim, razlike su se pojavile u kompletnom modelu, većinom u delu talasnih oblika koji odgovaraju vremenu prednjeg dela impulsne krive strujnog izvora. Ti rezultati dovode do zaključka da izostavljanje i zanemarivanje induktivne sprege (odnosno među-induktivnosti) igra značajnu ulogu u odzivu celog sistema. Tokom prelaznih procesa, izražen je veliki uticaj struje u jednom provodniku na druge provodnike koji su u induktivnoj sprezi. A po svojoj prirodi, svi elementi u tornju su induktivno spregnuti. Posebice struju atmosferskog pražnjenja karakteriše velika strmina (odn. impuls) na početku, koji zapravo znači veliku vrednost derivacije struje po vremenu. Taj činilac je vrlo bitan i ističe i doprinosi uticaju među-induktivnosti u kolu. Tako da, u konkretnom slučaju, među-induktivnost ima značajnu ulogu u rezultujućim raspodelama struja i prenaponima koji se pojavljuju u kolu.

7.3. Prednosti i nedostaci uprošćenog modelovanja

Svi delovi vetrogeneratora su modelovani uprošćeno, zbog usmerenosti ka posmatranju celokupnog modela i posmatranju željene rapodele bitnih veličina u celom modelu. Stoga, pojavile su se određene prednosti i nedostaci ovakvog pristupa. Neke od prednosti su: razumevanje modela i celokupne mreže kao i relativno lako primenjivanje prolaznih procesa; modeli se lako mogu prilagoditi različitim vrstama vetroturbina; relativno jednostavno prikupljanje podataka potrebnih za parametrizaciju modela; jednostavno manevrisanje modelima i ako je potrebno, jednostavna promena modela; moguća je uglavnom stabilna simulacija; moguće je posmatrati različite odzive u skladu sa promenom parametara; lako upoznavanje inženjera sa upotrebom modela ili celokupne simulacijske mreže.

U drugu ruku, sa pojednostavljenim modelovanjem, pojedini efekti nisu simulirani u potpunosti. Neki od njih su od većeg, neki od manjeg značaja kao što dobijeni rezultati pokazuju. Na primer, izostavljena induktivna sprega kablova je izazvala značajne posledice i promene u odzivu sistema. Karakteristična impedansa rotorske elise je računata na osnovu formule za izračunavanje impedanse provodne niti paralelne provodnoj površini iako je položaj u kome se injektira strujni impuls okomit na zemljinu površinu. Isto tako, karakteristična impedansa tornja (i kablova unutar) nije zavisna od visine, iako kapacitivnost sa visinom opada, a induktivnost raste.

Naposletku, posmatrani su samo električni delovi vetrogeneratora kao sistema. Nisu uzimani u obzir i razmatrani mehanički delovi sistema, niti bilo kakav vid konverzije mehaničke energije vetra u električnu energiju koju generiše generator.

8. ZAKLJUČAK

Rezultati rada pokazuju mogućnost modelovanja istih elemenata u Matlab/Simulink-u kao i u ATP/ETP-u, za specifičan slučaj vetrogeneratora tokom atmosferskog pražnjenja. Mogućnost zamene pojedinačnih delova modela, i na kraju celokupnog modela obezbeđuju mogućnost korišćenja softverskog alata Simulink za analizu sistema u kojima su od značaja tranzijentni procesi. Moguće ih je kombinovati i sa implementacijom drugih delova različitih sistema, kakvi su široko primenjivani u softverskom okruženju Simulink.

Kao bitan rezultat ističe se značaj i uticaj induktivne sprege u modelu. Prilikom izostavljanja međusobne induktivnosti tokom modelovanja dolazi do značajnijih promena, koje se ne bi smele zanemarivati.

9. LITERATURA

- [1] A. V. Rakov and A. M. Uman, "Lightning. Physics and effects", United Kingdom, Cambridge University Press, 2003.
- [2] H. Weifei, "Advanced Wind Turbine Technology", New York: Springer, 2018.
- [3] P. Hasse, J. Wiesinger and W. Zischank, "Handbuch für Blitzschutz und Erdung", Berlin, 2006.
- [4] F. Heidler, "Analytische Blitzstromfunktion zur LEMP-Berechnung", 18th ICLP Intern. Conf. on Lightning Protection, Munich, 1985.
- [5] H. K. H. LászlóPrikler, "ATPDRAW version 5.6 for Windows 9x/NT/2000/XP/Vista, Users' Manual", ATPDraw version 5.6, 2009.
- [6] "The MathWorks, Simulink® Getting Started Guide" United States, 2016.

Kratka biografija:



Milana Komosar rođena je u Prijedoru, Bosna i Hercegovina, 1994. god. Osnovne studije završila je na Fakultetu tehničkih nauka (FTN) iz oblasti Elektrotehnike i računarstva – Elektroenergetski sistemi 2018. god. Master studije, takođe na FTN na modulu Distribuirani elektroenergetski resursi završila je 2019. god. uz provela jedan semestar na TU Ilmenau (Nemačka).



Michael Rock rođen je 1971. god. Studirao je elektrotehniku na TU Ilmenau gde je diplomirao 1996. god., odnosno doktorirao 2001. god. Pre izbora u zvanje profesora na TU Ilmenau, radio je od 2009. do 2011. god. kao inženjer na projektu u kompaniji Dehn + Sohne.



Vladimir Katić rođen je 1954. god. u Novom Sadu. Doktorirao je na Univerzitetu u Beogradu 1991. god. Od 2002. god. je redovni profesor Univerziteta u Novom Sadu. Oblasti interesovanja su energetska elektronika, obnovljivi izvori energije, električna vozila i kvalitet električne energije.

BEZBEDONOSNI I INFORMACIONI SISTEM SOLARNIH ELEKTRANA SA PRORAČUNOM OSVETLJENJA**SECURITY AND INFORMATION SYSTEM OF SOLAR PLANTS WITH LIGHTING CALCULATION**

Aleksandra Stevanović, Vladimir Katić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu je predstavljen opis bezbedonosnih i informacionih sistema, koje solarne FN elektrane treba da poseduju sa konkretnim primerima informacionih i bezbedonosnih sistema. Izvršena je uporedna analiza četiri softverska alata za projektovanje osvetljenja i dat je proračun za jednu solarnu elektranu. Na kraju je data tehno-ekonomska analiza.

Ključne reči: Solarne FN elektrane, sigurnosni i bezbedonosni sistem, proračun osvetljenja

Abstract – In the paper description of security and information system for solar PV plants with examples of security and information systems is presented. Four softwares for lighting calculations are compared and lighting calculation for one solar plant is presented. Techno-economic analysis is given at the end.

Keywords: Solar plant, Security and Information system, lighting calculation

1. UVOD

Problem sigurnosti je aktuelna tema za solarne fotonaponske (FN) elektrane, kao i za druge distribuirane generatore. Ove elektrane se često nalaze na izolovanim područjima, koja su određena solarnim potencijalom, potrebnim prostorom i drugim pogodnostima, ali su time izložene značajnijim bezbedonosnim i operativnim rizicima. Postoji više rešenja za zaštitu ovakvih objekata, kao što su:

- video nadzor – postupak za prepoznavanje i procenu pokreta preko kamera i inteligentnog softvera, kao i aktiviranje alarma ako je potrebno.
- perimetrijska zaštita - zasnovana je na detekciji prisustva i kretanja ljudi unutar čuvanog prostora,
- ograđivanje električnom ogradom ili 3D panelima,
- sistemi za zaštitu pomoću mikrotalasa – jedinice ove zaštite sastoje se od predajnika i prijemnika koji su postavljeni jedan nasuprot drugom.

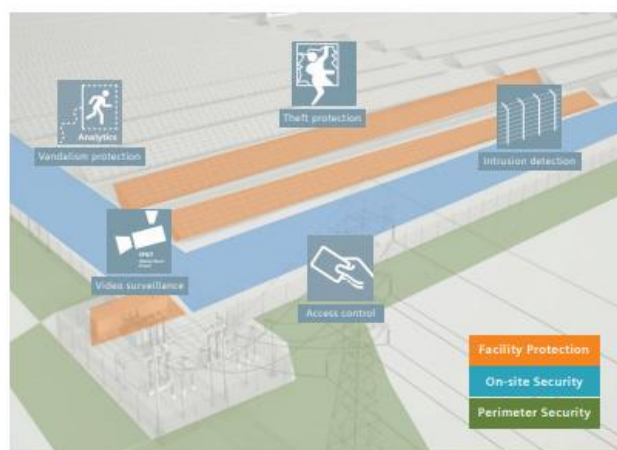
Na slici 1 data je ilustracija mogućih sistema za nadzor jedne solarne elektrane.

Pored sistema za zaštitu, potrebno je imati i informacioni sistem. Informacioni sistem prati performanse elektrane u cilju optimizacije proizvodnje i prevencije kvara. Sastoji se od senzora i mernih uređaja, koji prate i mere uslove

sredine i izlaze panela. Pored senzora, postoje komunikacijski uređaji koji su povezani sa SCADA sistemom mrežnog invertora, a preko interneta i sa računarom za nadzor i sladištenje podataka.

Bitna karika u bezbedonosnom sistemu jedne solarne FN elektrane je njeno spoljno osvetljenje. Spoljašnje osvetljenje je neophodno u obezbeđivanju površina i objekata od značaja u okviru postrojenja. Pod površinama od značaja podrazumevaju se FN paneli, transformator, objekat sa inverterima i objekat kontrolne sobe u kome se nalazi sistem za prikupljanje informacija sa senzora i mernih uređaja. Ako dođe do kvara bitno je osvetliti ove tačke solarne elektrane. Pored toga spoljašnje osvetljenje se koristi za unapređenje sigurnosti, jer iako se u okviru video nadzora koriste kamere sa infracrvenim reflektorima za noćno snimanje, osvetljenje površina daje jasniji i precizniji pogled. Postoji mogućnost korišćenja osvetljenja sa senzorima pokreta i napajanjem iz obnovljivih izvora tj. pametnog osvetljenja, čime se smanjuju troškovi i produžuje vek korišćenja.

U ovom radu je urađena uporedna analiza četiri softverska alata za dizajniranje i proračun osvetljenja, koje je neophodno za nadzor i obezbeđivanje solarne elektrane. Od ta četiri softvera, kao najpovoljniji izabran je Relux za proračun osvetljenja jedne solarne elektrane, a na kraju je data tehno-ekonomska analiza.



Slika1. Primer sistema za nadzora jedne solarne elektrane [1]

2. UPOREDNA ANALIZA SOFTVERSKIH ALATA ZA PROJEKTOVANJE OSVETLJENJA

Iako postoji više desetina programa za dizajniranje i proračun osvetljenja, svi su zasnovani na dva matema-

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Katić, red. prof.

tička algoritma: algoritam radiostatičnosti i algoritam zračenja. Dialux, Relux, AGI32 i Radiance su četiri najpopularnija softverska alata i nad njima je izvršena uporedna analiza. Svaki od ovih programa ima mane i vrline u odnosu na ostale. AGI32 daje najtačnije proračune, ali nije besplatan. Radiance, takođe daje izuzetno tačne proračune, ali nema korisnički pogodan interfejs. Dialux i Relux daje proračune slične tačnosti, ali je Relux za nijansu brži. Vreme trajanja simulacija u ovim programima dato je u tabeli 1. Relux je najbrži, a pored brzine, daje i poprilično tačne rezultate u svim situacijama. Pokazao se kao najpogodniji kada uvažimo njegov interfejs, izlazne izveštaje i podatke, efikasnost, podršku, koju pruža korisnicima i modelovanje u samom programu.

Tabela1. Vreme trajanja simulacija u četiri analizirana softverska alata za simulaciju osvetljenja [2]

Vreme trajanja simulacije	DIALux [s]	Relux [s]	AGI32 [s]	Radiance [s]
Dnevno osvetljenje	70	10	24	193
Dnevno i veštačko osvetljenje	90	10	35	225
Veštačko osvetljenje	8	5	10	53

3. OPIS RADA U RELUX-U I PRIMENA ZA DIZAJNIRANJE OSVETLJENJA SOLARNE FN ELEKTRANE

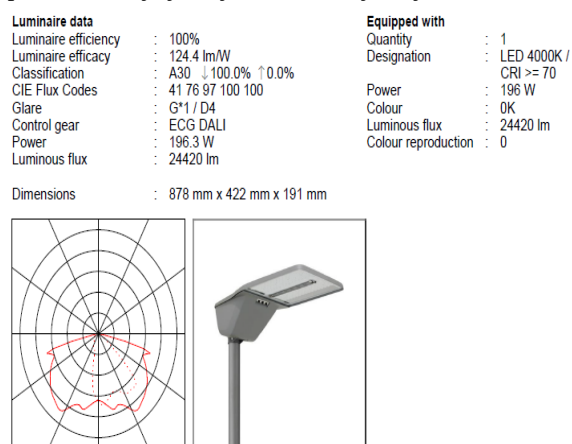
3.1 Postavka problema

Za solarnu FN elektranu, koja će poslužiti za dizajniranje spolnog osvetljenja uRelux-u uzeto je solarno postrojenje u Stolcu snage 150 kW, koje zauzima 3.650 m², ima 648 FN panela, raspoređenih u 4 celine (grupe). Svaka celina ima 9 nizova, a svaki niz sačinjen je od 18 FN panela. Pored panela, na zemljištu gde se nalazi solarna elektrana, postoji i objekat za nadzor i razvodno postrojenje za povezivanje na mrežu.

3.2 Projektovanje osvetljenja solarne FN elektrane

Relux ima mogućnost da se unese CAD fajl enterijera, eksterijera, ulice ili tunela i uradi proračun osvetljenja ili da se crta posmatrani prostor direktno u programu. Za potrebe ovog projekta, nacrtan je plan postrojenja u Aecosim softveru. Aecosim je program za crtanje ekvivalentan Autodesk-ovom AUTOCAD-u i Revit-u. Nakon što je nacrtana solarna farma, potrebno je izvesti crtež u datoteku sa ekstenzijom .dwg jer je to ekstenzija koja može da se uveze u Relux. Sledeći korak je uvoz CAD datoteke u program za dizajniranje osvetljenja. S obzirom da se pomoću Relux-a može modelovati unutrašnje i spoljašnje osvetljenje potrebno je pri samom otvaranju izabrati koji tip osvetljenja se modeluje. Zatim se unose osnovni podaci koji će biti ispisani u izveštaju. To su: broj projekta, naručilac/kupac projekta, ime projektantske firme, datum i kratak opis projekta. Posle ispunjavanja ovih podataka, uvozi se .dwg fajl i bira se koordinatni početak i razmera. Unesen je fajl u obliku 2D crteža. Zbog toga je potrebno sve objekte koji bi mogli bacati senku nacrtati u 3D obliku.

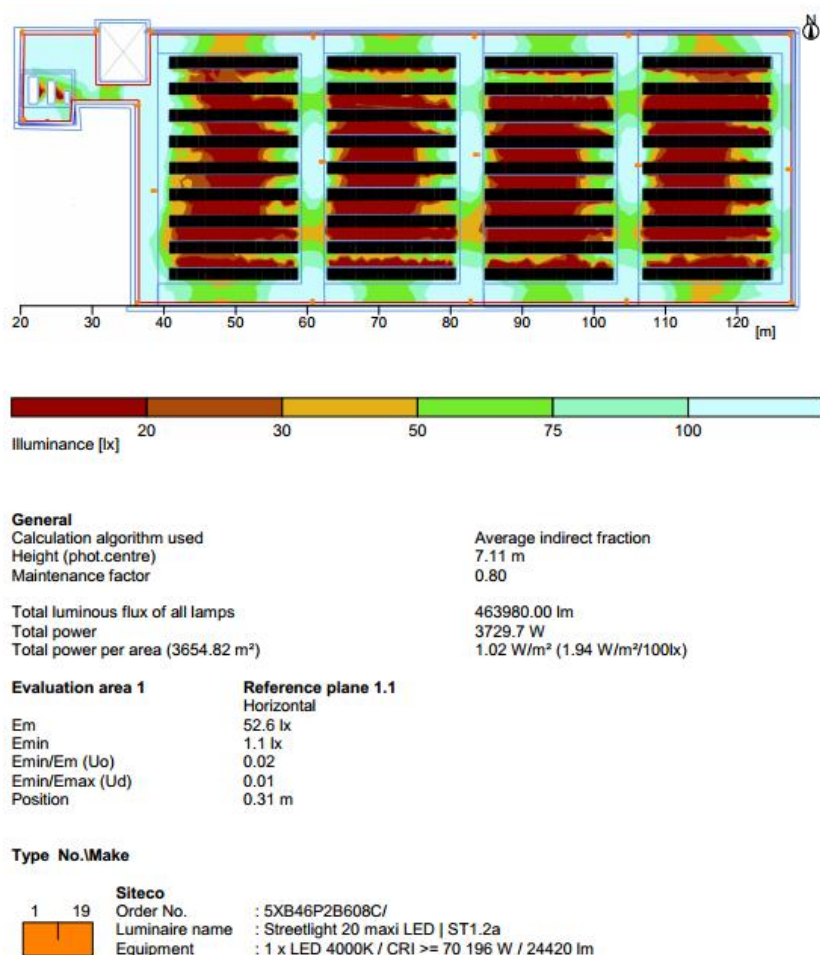
Pomoću opcija iz kartice INSERT na 2D planu nacrtana je ograda koja se nalazi oko postrojenja, objekat koji se nalazi u okviru solarne elektrane i razvodno postrojenje. To je nacrtano jer su to objekti od značaja za osvetljenje zbog bacanja senke i smanjivanje osvetljenosti područja. Sledeći korak je izbor sijalica. Relux pruža korisniku veliku bazu podataka sa više od 80 proizvođača. Moguće je preuzeti sijalicu sa zvaničnog sajta proizvođača i uvesti je u program. Pored toga, moguće je pristupiti online bazi podataka, izabrati sijalicu i uvesti je. Sijalice se unose u projekat pomoću kartice Products iz linije alatki. Kada se uvezu, u Relux-u se vide detaljni podaci o datoj sijalici. Za potrebe ovog projekta, izabrana je LED sijalica proizvođa SITECO. Podaci o sijalicama se nalaze na slici 2. Na slici vidimo kolika je dimenzija reflektora, kolika je njegova efikasnost, koliko lumena daje, kakva je usmerenost svetlosti koju daje reflektor, koliko snage mu je potrebno, koje je boje svetlost koju daje reflektor.



Slika 2. Podaci o izabranoj sijalici [3]

Kada se rasporede sijalice na željene pozicije i visinu, radi se proračun pomoću opcije "calculate" iz linije alatki. Moguće je podesiti faktor održavanja, visinu, nagibni ugao sijalice. Raspored sijalica i rezultat osvetljenje koje daju dat je na slici 3.

Na slici 3 se vidi da sve površine, koje se nalaze u okolini ograde i svi putevi, koji se nalaze između FN panela imaju osveljenost preko 100 luksa, što daje dovoljnu vidljivost pri pokušaju provale ili pri eventualnom kvaru. Sve sijalice se nalaze na stubu visine 7 m sa uglom nagiba do 10 stepeni. Taj ugao ne sme biti veći zbog insekata koji bi udarali o svetleći deo reflektora i tako smanjivali svetlosnu moć. Uvažen je i faktor održavanja u vrednosti 0,8 zbog toga što snaga sijalice opada sa vremenom rada. Snaga koja je potrebna svim sijalicama je 3,7 kW. Sve ove informacije i grafici se mogu odštampati u PDF formatu u formi izveštaju. Izveštaj može imati sadržaj, podatke o sijalicama, 2D/3D pogleda sa osvetljenjem i bez i dr. Moguće je izabrati šta će se sve prikazati u izveštaju u zavisnosti od potreba korisnika, projekta i kupca. U Relux-u postoji mogućnost dinamičkog planiranja. Pomoću ove opcije se menjaju pogledi na 3D sliku scene koja se modelira. Slika je prikazana zajedno sa sijalicama i veštačkim osvetljenjem koje one daju.



Slika 3. Raspored osvetljenja i osnovne informacije o osvetljenju.

4. TEHNO-EKONOMSKA ANALIZA

Da bi se uradila tehno-ekonomska analiza, potrebno je odrediti (proceniti) troškove, koji se odnose na kupovinu i ugradnju svih komponenti razmatranog sistema, kao i troškove vezane za održavanje sistema.

Prva komponenta, koja će se posmatrati je SUNNY WEBBOX, uređaj koji košta 560 €. Na njega se može priključiti 10 invertora, koliko postoji na solarnoj elektrani.

Za video nadzor ove solarne PV elektrane jeodabrano 25 8Mp bullet vodootpornih kamera sa ICR senzorom dometa 80 m. Detektuju upad u zonu, prelazak preko zamišljene linije i lice. Zaštita kamere od spoljašnjih uslova je IP67. Cena jedne kamere je 190 €, a ukupna cena je 4.750 €. Pored toga, potrebna su 2 DVR čitača sa po 16 izlaza i sa kablovima za napajanje i razdlenicima po ceni od 225 €/kom. Ukupna cena video nadzora je 5.200 €.

Za ograđivanje postrojenja je korištena 3D panelna ograda. Svi ogradni elementi su izrađeni od toplo pocinkovanog čelika i plastificirani poliesterom. Panel dimenzije 2500 mm x 2030 mm ima cenu od 26,3 €/kom. Debljina žice je 4 mm. Pored panelnih ograda, potrebno je ugraditi i noseće stubove, koji su takođe pocinkovani i plastificirani. Stubovi dimenzije 50 mm x 50 mm x 2050 mm imaju cenu od 14,7 €/kom. Za ograđivanje (rad) je potrbno platiti još 5.945 €.

Na kraju, korišteno je 19 LED reflektora. Cena jednog reflektora je 600 €. Sa ugradnjom pametnog sistema i napajanja preko sopstvenog solarnog panela, te cene idu i do 1.000 €. Ako se uzmu klasične LED sijalice cena za 19

sijalica bi bila 11.400€. Kada se sve sabere, troškovi informacionog sistema i bezbedonosnog sistema su 23.105 €.

S obzirom da solarna elektrana ima 648 panela i 10 invertora, a da je prosečna cena FN panela oko 200 €, invertora 2.200 €, te uz dodate troškove za kablove i razvodno postrojenje, ukupna ulaganja u FN elektranu se mogu proceniti na oko 170.000 – 180.000 €.

Sistem za zaštitu od krađe i informacioni sistem je oko 11% od ukupnog ulaganja u elektranu, što je s obzirom na vrednosti imovine (poseda i opreme) i koji se treba nadgledati i sačuvati, investicija u koju vredi uložiti. U tabeli 2 se nalazi jedinična cena i količina opreme koju treba uložiti u sigurnosni i informacioni sistem solarne elektrane.

Tabela 2. Troškovi za informacioni i bezbedonosni sistem

Vrsta opreme	Jedinična cena	Količina	Ukupna cena
SUNNY WEBBOX komunikacijski uređaj	560€	1	560€
HIKVISION kamera za nadzor	190€	25	4750€
DVR čitač	225€	2	450€
3D panelna ograda	41€	145	5945€
LED reflektor siteco	600€	19	11400€

5. ZAKLJUČAK

Zbog velikih početnih ulaganja i izolovanosti solarnih farmi, čija je pozicija obično uslovljena mestom koje je pogodno za građenje, potrebno je pronaći adekvatan sistem zaštite i upravljanja.

Što se tiče informacionog i upravljačkog dela, obično se koriste informacioni uređaji istog proizvođača ili drugog proizvođača koji su kompatibilni sa invertorima na solarnoj farmi. Ti uređaji stalno prikupljaju podatke i šalju ih osobama zaduženim za nadzor. Ulaganje u ove sisteme je isplativo jer pomoću njih možemo otkriti grešku u radu neke komponente solarne elektrane i reagovati u što kraćem roku kako bi se izbegli eventualni gubici zbog nepravilnog rada. Još jedna prednost ulaganja u informacione sisteme je mogućnost daljinske promene parametara svih invertora elektrane.

Bezbedonosni sistemi su bitan faktor u zaštiti solarnog postrojenja i njegovih objekata. Koriste se razne vrste ograda sa perimetrijskim zaštitama, te električne ograde visokih voltaža sa visokim stepenom odvratanja. Moguće je korišćenje sistema kartica za ograničavanje pristupa koje obezbeđuju velike sistem od više celina od neovlašćenog upada.

Pored ograđivanja farmi, neizostavan deo zaštite je video nadzor koji može biti pomoću analognog i IP video nadzora. IP video nadzor predstavlja sistem video nadzora kod kog je moguće pregledati snimke ili gledati uživo snimak preko mreže (LAN ili Internet). To je velika prednost jer nam daje mogućnost mobilnosti. Iako je IP video nadzor skuplji od analognog, IP video nadzor je bolje rešenje i isplativije na duže staze. Zaštite koje se ređe koriste jesu zaštite mikrotalasima i infracrvenim zracima koji rade na principu predajnik-prijemnik. Zbog mogućnosti nepravilnog rada pri nepovoljnim vremenskim uslovima i potrebe za izborom kvalitetnijih komponenti koji su skuplje, ova zaštita se koristi kada postoji potreba za visokim nivoom sigurnosti. U takvim situacijama se pored video nadzora i ograđivanja, koriste i visokokvalitetna zaštita mikrotalasima. Različite kombinacije ovih zaštita se koriste u skladu sa potrebama terena i ekonomskim mogućnostima investitora. Najčešće se koristi kombinacija električne ograde i video nadzora.

Za proračun osvetljenja, koriste se program za dizajniranje osvetljenja kojih ima na tržištu preko 30 od različitih kompanija. S obzirom na potrebe ovog projekta i uvažavajući osobine ovih programa, izabrana su 4 programa: Relux, DIALux, AGI32 i Radiance za uporednu analizu kako bi se izabrao najpogodniji program za dizajniranje osvetljenja solarnog postrojenja. AGI32 se ispostavio kao program koji daje najtačnije rezultate, ali je zbog plaćanja licence za korišćenje, bio nepogodan za ovaj proračun.

Radiance je takođe davao izuzetno tačne rezultate u proračunima po istraživanjima, ali je zbog interfejsa koji je komplikovan za korišćenje i "open source" koncepta isključen iz užeg izbora. Zbog toga što je odnos tačnosti rezultata i brzine obrade podataka i simulacije bolji kod Relux-a u odnosu na DIALux, Relux je program koji je izabran za dizajniranje osvetljenja solarne elektrane.

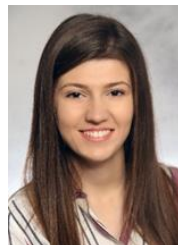
Na kraju rada, detaljno je objašnjen postupak rada u Relux-u, od uvoza crteža elektrane iz Aecosim-a u Relux preko crtanja objekata u 3D-u koji bi mogli bacati senku na osvetljena područja do izbora sijalica, kalkulacije i izrade izveštaja u .pdf format. Izabrano je 19 LED sijalica koje troše po 196W sa solarnim panelom na svom stubu.

Da zaključimo, solarno ulično osvetljenje nije usvojeno do sada kao pouzdana zamena tradicionalnom. Danas, uz pomoć pametne tehnologije, u mogućnosti smo da pretvorimo solarno ulično osvetljenje u standardizovano rešenje za osvetljenje, demistifikujući njegovu lošu reputaciju iz prošlosti. Investicija u pametno solarno ulično osvetljenje je zaista vredna i u budućnosti će se isplatiti.

6. LITERATURA

- [1] <https://www.downloads.siemens.com/-downloadcenter/Download.aspx?pos=download&fct=getasset&id1=A6V10594616>
- [2] https://issuu.com/peter.byrne1000/docs/dissertation_-_peter_byrne_-_publis
- [3] <https://reluxnet.relux.com/en/>

Kratka biografija:



Aleksandra Stevanović rođena je 1994. godine u Bijeljini. Fakultet tehničkih nauka, studijski program Energetika, elektronika i telekomunikacije upisala je školske 2013/2014 godine. Na studijama se opredelila za smer Elektroenergetski sistemi i diplomirala 2018. god. Master studije je upisala školske 2018/2019. god. na studijskom programu Energetika, elektronika i telekomunikacije, modul Elektroenergetika –Distribuirani elektroenergetski resursi (MAS).



Vladimir Katić rođen je 1954. godine u Novom Sadu. Doktorirao je na Univerzitetu u Beogradu 1991. godine. Od 2002. godine je redovni profesor Univerziteta u Novom Sadu. Na Fakultetu tehničkih nauka sada je Prodekan za finansije i razvoj (od 1998 god.). Oblasti interesovanja su energetska elektronika, obnovljivi izvori električne energije, električna vozila i kvalitet električne energije.

MODELOVANJE KORELACIJE MERENJA U ESTIMACIJI STANJA**MODELLING OF MEASUREMENTS CORRELATION IN THE STATE ESTIMATION**Milena Lisov, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – U ovom radu opisan je WLS algoritam estimacije stanja. Na primeru je urađeno modelovanje korelacije merenja sa digitalnih uređaja i pseudo merenja. Na kraju su rezultati modelovanja korelacije merenja analizirani.

Ključne reči: WLS estimacija stanja, korelacija merenja, modelovanje korelacije merenja, digitalni uređaji, pseudo merenja

Abstract – In this paper a WLS algorithm for the state estimation is described. The example is used for modelling of correlation of measurements from digital devices and for modelling correlation of pseudo-measurements. At the end, obtained results are analyzed.

Keywords: WLS state estimation, correlation of measurements, modelling of correlation of measurements, digital devices, pseudo-measurements

1. UVOD

Poznavanje aktuelnog režima elektroenergetskog sistema (EES) je osnova za sve proračune vezane za analizu, upravljanje, eksploataciju, itd. Zbog toga je estimacija stanja osnovna energetska funkcija za najveći broj elektroenergetskih proračuna. Ova funkcija vrši konvertovanje telemetrisanih (aktuelnih) merenja iz realnog vremena i pseudo merenja u pouzdan vektor stanja. Pored toga, funkcija vrši i rekonstrukciju celokupnog režima sistema na osnovu vektora stanja. Kada su promenljive stanja poznate, za datu konfiguraciju i definisane parametre sistema, mogu se jednoznačno proračunati i sve druge systemske promenljive od interesa. Poznavanje stanja EES-a u nekom diskretnom vremenskom trenutku, ali ne i način na koji je sistem dospao u taj režim, estimaciju stanja čini statičkom. Izučavanje statičkih pojava u sistemu podrazumeva upotrebu modela zasnovanih na algebarskim jednačinama. Može se reći da statička estimacija stanja predstavlja on-line proračune vektora promenljivih stanja i ostalih systemskih promenljivih, na bazi periodičnih merenja iz sistema, sa ciklusom ponavljanja od nekoliko sekundi do nekoliko minuta.

Najčešće korišćeni algoritam u statičkoj estimaciji stanja jeste metod minimuma sume otežanih (ponderisanih) kvadrata reziduala merenja (Weighted Least Squares Method-WLS metod).

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Andrija Sarić, red. prof.

Korelacija merenja predstavlja međusobnu zavisnost dva merenja, bez obzira na vrstu i tip merenja. Ona se modeluje kroz vandijagonalne elemente matrice kovarijansi merenja, pa samim tim uvažavanje korelacije merenja u estimaciji stanja narušava dijagonalnost matrice kovarijansi merenja. Obično se korelacija između merenja sa digitalnih uređaja i između pseudo merenja zanemaruje u cilju optimizacije računarske efikasnosti prilikom proračuna. U ovom radu će na primeru biti istražen uticaj modelovanja korelacije merenja u proračunima statičke estimacije stanja.

2. VARIJANSA MERENJA I TEŽINSKI FAKTOR

Merenjima se pridružuju varijanse, odnosno težinski faktori sa kojima ona ulaze u dalji proračun. Varijansa, tj. težinski faktor merenja se definiše preko matrice kovarijansi merenja $\mathbf{R}$ preko relacije [1]:

$$\mathbf{R} = \text{cov}\{\mathbf{e}\} = E\{\mathbf{e}\mathbf{e}^T\} = \text{diag}\{\sigma_m^2\}, \quad (1)$$

gde je $\mathbf{e}$ vektor grešaka merenja, σ_m^2 je varijansa m -tog merenja, a $m = 1, 2, \dots, M$.

Druga forma prethodne matrice je dijagonalna matrica težinskih faktora $\mathbf{W}$:

$$\mathbf{W} = \text{diag}\{w_m\} = \mathbf{R}^{-1} = \text{diag}\left\{\frac{1}{\sigma_m^2}\right\}. \quad (2)$$

Treba naglasiti da su matrice $\mathbf{R}$ i $\mathbf{W}$ uvedene kao dijagonalne matrice. To je slučaj samo kada je korelacija između različitih merenja zanemarena. Kada se korelacija merenja ne zanemaruje ove matrice nisu dijagonalne, kao što će to biti slučaj u ovom radu.

3. WLS ALGORITAM

Problem statičke estimacije stanja moguće je definisati 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 $\mathbf{x}$), pri čemu se svakoj razlici (koja se naziva rezidual merenja) mora dodeliti odgovarajuća težina.

Početna jednačina za algoritam WLS estimacije stanja je [2]:

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_M \end{bmatrix} = \begin{bmatrix} h_1(x_1, x_2, \dots, x_n) \\ h_2(x_1, x_2, \dots, x_n) \\ \vdots \\ h_M(x_1, x_2, \dots, x_n) \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_M \end{bmatrix} = \mathbf{h}(\mathbf{x}) + \mathbf{e}. \quad (3)$$

Vektor $\mathbf{z}$ za M vrednosti merenja je [2]:

$$\mathbf{z}^T = [z_1 \quad z_2 \quad \dots \quad z_M]. \quad (4)$$

Vektor $\mathbf{h}$ sadrži:

$$\mathbf{h}^T = [h_1(\mathbf{x}) \quad h_2(\mathbf{x}) \quad \dots \quad h_M(\mathbf{x})]. \quad (5)$$

Kao što se vidi, vektor $\mathbf{h}$ sadrži M nelinearnih funkcija $h_i(\mathbf{x})$ koje pretpostavljene vrednosti merenja i vezuju za vektor stanja $\mathbf{x}$ koji sadrži n promenljivih.

$$\mathbf{x}^T = [x_1 \ x_2 \ \dots \ x_n]. \quad (6)$$

Vektor $\mathbf{e}$ predstavlja vektor grešaka merenja koji se sastoji od sledećih elemenata:

$$\mathbf{e}^T = [e_1 \ e_2 \ \dots \ e_M]. \quad (7)$$

Kriterijumska funkcija se predstavlja sledećom relacijom [2]:

$$J(\mathbf{x}) = \sum_{i=1}^M \frac{(z_i - h_i(\mathbf{x}))^2}{R_{ii}} = [\mathbf{z} - \mathbf{h}(\mathbf{x})]^T \mathbf{R}^{-1} [\mathbf{z} - \mathbf{h}(\mathbf{x})]. \quad (8)$$

Cilj WLS estimatora je minimizacija prethodno date kriterijumske funkcije:

$$\min_{\mathbf{x}} \{J(\mathbf{x}) = [\mathbf{z} - \mathbf{h}(\mathbf{x})]^T \mathbf{R}^{-1} [\mathbf{z} - \mathbf{h}(\mathbf{x})]\}. \quad (9)$$

U ovom radu je za rešavanje optimizacionog kriterijuma korišćen metod normalnih jednačina.

Estimacija vektora promenljivih stanja se dobija iterativnim rešavanjem sledećeg sistema linearizovanih jednačina [2]:

$$\mathbf{G}(\mathbf{x}^{(k)}) \Delta \mathbf{x}^{(k)} = \mathbf{H}^T(\mathbf{x}^{(k)}) \mathbf{R}^{-1} (\mathbf{z} - \mathbf{h}(\mathbf{x}^{(k)})) \quad (10)$$

$$= \mathbf{H}^T(\mathbf{x}^{(k)}) \mathbf{R}^{-1} \Delta \mathbf{z}^{(k)}; k = 0, 1, 2, \dots$$

$$\Delta \mathbf{x}^{(k)} = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}, \quad (11)$$

gde je:

$\mathbf{G}(\mathbf{x}^{(k)}) = \mathbf{H}^T(\mathbf{x}^{(k)}) \mathbf{R}^{-1} \mathbf{H}(\mathbf{x}^{(k)})$ – matrica pojačanja u k -toj iteraciji;

$\mathbf{H}(\mathbf{x}^{(k)})$ – Jakobijan matrica u k -toj iteraciji;

$\Delta \mathbf{z}^{(k)} = \mathbf{z} - \mathbf{h}(\mathbf{x}^{(k)})$ – priraštaj vektora merenja u k -toj iteraciji;

$\Delta \mathbf{x}^{(k)}$ – korekcija vektora promenljivih stanja u k -toj iteraciji.

Iterativni postupak se završava kada je zadovoljena sledeća nejednakost:

$$\max\{\Delta \mathbf{x}^{(k)}\} \leq \varepsilon, \quad (12)$$

gde je ε kriterijum konvergencije.

Kada je skup merenja dovoljan i pravilno distribuiran (observabilan sistem), tada je Jakobijan matrica $\mathbf{H}(\mathbf{x})$ punog ranga, pa je matrica pojačanja $\mathbf{G}(\mathbf{x})$ nesingularna matrica, tako da je sigurno obezbeđena numerička stabilnost proračuna. S obzirom da je matrica $\mathbf{H}(\mathbf{x})$ retka, matrica pojačanja $\mathbf{G}(\mathbf{x})$ takođe je retka (ali ipak gušća od Jakobijan matrice).

4. MODELOVANJE KORELACIJE MERENJA

Vandijagonalni elementi matrice kovarijansi grešaka merenja definišu se na sledeći način [3]:

$$\sigma_{ij} = \rho_{ij} \sigma_i \sigma_j, \quad (13)$$

gde su:

σ_{ij} – međusobna varijansa i -tog i j -tog merenja, $i, j = 1, 2, \dots, M$;

ρ_{ij} – faktor korelacije i -tog i j -tog merenja, $i, j = 1, 2, \dots, M$;

σ_i – varijansa i -tog merenja, $i = 1, 2, \dots, M$;

σ_j – varijansa j -tog merenja, $j = 1, 2, \dots, M$;

Vrednost faktora korelacije nezavisno od estimacije stanja može biti u sledećem opsegu:

$$-1 < \rho_{ij} < 1. \quad (14)$$

Ako postoji korelacija između merenja i i j , onda ta korelacija postoji i između merenja j i i . Kada je vrednost faktora korelacije pozitivna, radi se o pozitivnoj korelaciji merenja, u suprotnom korelacija između merenja je negativna.

Matrica kovarijansi grešaka merenja $\mathbf{R}$ sa uključenom korelacijom merenja ima sledeći zapis [4]:

$$\mathbf{R} = \begin{bmatrix} \sigma_1^2 & \dots & \rho_{1j}\sigma_1\sigma_j & \dots & \rho_{1M}\sigma_1\sigma_M \\ \vdots & \sigma_2^2 & & & \vdots \\ \rho_{i1}\sigma_i\sigma_1 & & \ddots & & \vdots \\ \vdots & & & \ddots & \vdots \\ \rho_{M1}\sigma_M\sigma_1 & \dots & \dots & \dots & \sigma_M^2 \end{bmatrix} \quad (15)$$

Merenja sa digitalnih uređaja imaju manju varijansu od pseudo merenja. Ova činjenica se reflektuje većom tačnošću merenja dobijenih sa digitalnih mernih uređaja od istorijskih podataka [4].

Treba napomenuti da situacija kada je $\rho_{ij} = 1.0$ uzrokuje singularnost matrice kovarijansi grešaka merenja $\mathbf{R}$, odnosno uzrokuje divergenciju estimacije stanja [5]. Zbog toga je vrednost 0.99 odabrana za faktor korelacije potpuno korelisanih merenja.

Gledajući relaciju (15), jasno je da prisustvo korelacije smanjuje retkost matrice kovarijansi grešaka merenja $\mathbf{R}$, što ima za posledicu smanjenje brzine inverzije matrice pojačanja $\mathbf{G}$. Zbog toga je bitno razumeti kada je relevantno uključiti korelaciju u proračune radi poboljšanja performansi estimacije, a kada se, sa druge strane, estimacija može održavati dovoljno tačnom bez degradacije brzine procesa estimacije [4].

Radi određivanja uticaja uključivanja korelacije u proračune vrši se proračun korena srednje vrednosti kvadrata grešaka merenja (Root Mean Square Errors - RMSE). Formula za proračun RMSE glasi [4]:

$$RMSE_y = \sqrt{\frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{N}}, \quad (16)$$

gde je y_i tačna vrednost veličine, $\hat{y}_i$ estimirana vrednost veličine, a N je broj iteracija prilikom proračuna estimacije stanja. Kako je rečeno u [3] vrednosti promenljivih koje su dobijene proračunom tokova snaga za poznatu konfiguraciju i parametre mreže se smatraju tačnim vrednostima.

4.1 Korelacija merenja sa digitalnih uređaja

U analizi koja je urađena u ovom radu pretpostavlja se da digitalni uređaji koji se nalaze u mreži mere moduo napona i tok aktivne i reaktivne snage na dve različite grane.

Digitalni merni uređaji obično kao izlazne veličine obezbeđuju merenja napona i snage počinjući od direktnih merenja modula napona i struje i razlike faznog ugla napona i struje. Na osnovu ovih merenja indirektno se proračunavaju tokovi aktivnih i reaktivnih snaga [4].

Merenja toka aktivne P i reaktivne snage Q mogu se izraziti kao funkcija modula napona U , modula struje I i razlike faznog ugla napona i struje φ kao:

$$P = P(V, I, \varphi) = VI \cos \varphi, \quad (17)$$

$$Q = Q(V, I, \varphi) = VI \sin \varphi. \quad (18)$$

Grupisanjem jednačina (17) i (18) u vektorsku funkciju dobija se:

$$\mathbf{f}(V, I, \varphi) = \begin{bmatrix} V \\ P(V, I, \varphi) \\ Q(V, I, \varphi) \end{bmatrix}. \quad (19)$$

Ukoliko se varijanse modula napona, modula struje i faznog ugla obeleže sa $\sigma_V^2, \sigma_I^2, \sigma_\varphi^2$, respektivno, matrica kovarijansi merenja može se izračunati na sledeći način [4]:

$$E[\Delta \mathbf{z}_{vpq} \Delta \mathbf{z}_{vpq}^T] = \begin{bmatrix} \sigma_V^2 & \sigma_{VP} & \sigma_{VQ} \\ \sigma_{PV} & \sigma_P^2 & \sigma_{PQ} \\ \sigma_{QV} & \sigma_{QP} & \sigma_Q^2 \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathbf{f}}{\partial \mathbf{z}_{v\varphi i}} \end{bmatrix} \times \begin{bmatrix} \sigma_V^2 & & \\ & \sigma_\varphi^2 & \\ & & \sigma_I^2 \end{bmatrix} \times \begin{bmatrix} \frac{\partial \mathbf{f}^T}{\partial \mathbf{z}_{v\varphi i}} \end{bmatrix}, \quad (20)$$

gde su:

$\mathbf{z}_{vpq} = [\mathbf{V} \ \mathbf{P} \ \mathbf{Q}]^T$ – zavisni vektor merenja;

$\mathbf{z}_{v\varphi i} = [\mathbf{V} \ \varphi \ \mathbf{I}]^T$ – nezavisni vektor merenja;

$\frac{\partial \mathbf{f}}{\partial \mathbf{z}_{v\varphi i}}$ – Jakobijanski vektorske funkcije (19).

Vektorska funkcija $\mathbf{f}$ povezuje nezavisni vektor merenja $\mathbf{z}_{v\varphi i}$ sa zavisnim vektorom merenja $\mathbf{z}_{vpq}$. Njen Jakobijanski $\frac{\partial \mathbf{f}}{\partial \mathbf{z}_{v\varphi i}}$ se zapisuje na sledeći način:

$$\frac{\partial \mathbf{f}}{\partial \mathbf{z}_{v\varphi i}} = \begin{bmatrix} \frac{\partial V}{\partial V} & \frac{\partial V}{\partial \varphi} & \frac{\partial V}{\partial I} \\ \frac{\partial P}{\partial V} & \frac{\partial P}{\partial \varphi} & \frac{\partial P}{\partial I} \\ \frac{\partial Q}{\partial V} & \frac{\partial Q}{\partial \varphi} & \frac{\partial Q}{\partial I} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ I \cos \varphi & -VI \sin \varphi & V \cos \varphi \\ I \sin \varphi & VI \cos \varphi & V \sin \varphi \end{bmatrix}. \quad (21)$$

Uvrštavanjem (21) u jednačinu (20) i njenim proračunom dobijaju se varijanse aktivne i reaktivne snage u sledećem obliku:

$$\sigma_P^2 = \sigma_V^2 I^2 \cos^2 \varphi + \sigma_\varphi^2 V^2 I^2 \sin^2 \varphi + \sigma_I^2 V^2 \cos^2 \varphi, \quad (22)$$

$$\sigma_Q^2 = \sigma_V^2 I^2 \sin^2 \varphi + \sigma_\varphi^2 V^2 I^2 \cos^2 \varphi + \sigma_I^2 V^2 \sin^2 \varphi. \quad (23)$$

Međusobne varijanse napona i aktivne snage, napona i reaktivne snage, aktivne snage i reaktivne snage su:

$$\sigma_{VP} = \sigma_V^2 I \cos \varphi, \quad (24)$$

$$\sigma_{VQ} = \sigma_V^2 I \sin \varphi, \quad (25)$$

$$\sigma_{PQ} = \frac{1}{2} (\sigma_V^2 I^2 \sin 2\varphi - \sigma_\varphi^2 V^2 I^2 \sin 2\varphi + \sigma_I^2 V^2 \sin 2\varphi). \quad (26)$$

Kako se nesigurnost obično izražava u relativnim vrednostima, jednostavnije je varijanse koje se izračunavaju relacijama (22) i (23) podeliti sa kvadratom odgovarajuće merene vrednosti. Na taj način se dobija:

$$\frac{\sigma_P^2}{P^2} = \frac{\sigma_V^2}{V^2} + \frac{\sigma_I^2}{I^2} + \sigma_\varphi^2 \tan^2 \varphi, \quad (27)$$

$$\frac{\sigma_Q^2}{Q^2} = \frac{\sigma_V^2}{V^2} + \frac{\sigma_I^2}{I^2} + \sigma_\varphi^2 \cot^2 \varphi, \quad (28)$$

a međusobne varijanse iz relacija (24)-(26) mogu se napisati u sledećem obliku:

$$\frac{\sigma_{VP}}{VP} = \frac{\sigma_V^2}{V^2}, \quad (29)$$

$$\frac{\sigma_{VQ}}{VQ} = \frac{\sigma_V^2}{V^2}, \quad (30)$$

$$\frac{\sigma_{PQ}}{PQ} = \frac{\sigma_V^2}{V^2} + \frac{\sigma_I^2}{I^2} - \sigma_\varphi^2. \quad (31)$$

4.2 Korelacija pseudo merenja

Pseudo merenja predstavljaju pretpostavljene (planske) vrednosti nekih veličina, kao što su, na primer, prognozirana opterećenja u čvorovima. Pseudo merenja su „proizvedeni” podaci koji se baziraju na arhivskim podacima ili proceni dispečera, odnosno analitičara. Ona se obično uvode sukcesivno (jedno po jedno) da bi se neobservabilni delovi mreže učinili observabilnim. Ova merenja se najčešće dobijaju iz prognoze opterećenja čvorova mreže i čine ih injektirane snage u čvorovima.

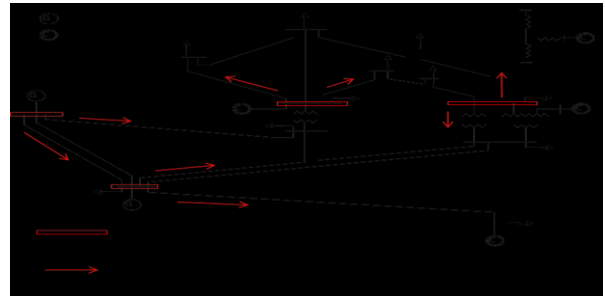
Korelacija pseudo merenja može biti prisutna između injektiranih snaga potrošačkih čvorova. Ovaj vid korelacije naziva se inter-node korelacija. Nju može uzrokovati nekoliko faktora, kao na primer sličnost potrošnje, vremenski uslovi, cena energije ili operacije upravljanja mrežom [6]. Takođe, inter-node korelacija može biti prisutna i između injektiranih snaga generatora u mreži.

Korelacija može da se pretpostavi i između injektiranja aktivne i reaktivne snage jednog čvora uzimajući u obzir čestu pretpostavku o velikom stepenu poznavanja faktora snage za HV (High Voltage) i MV (Medium Voltage) potrošače. U ovom slučaju reč je o intra-node korelaciji [4].

Ukoliko su prisutni različiti tipovi korelacije svi indirektni unakrsni faktori korelacije se moraju uzeti u obzir prilikom formiranja matrice kovarijansi merenja. Na primer, ako su injektirane snage dva potrošačka čvora korelisane, kao i aktivne i reaktivne snage svakog čvora, korelacija nastaje i između aktivne snage jednog čvora i reaktivne snage drugog čvora [6].

5. REZULTATI MODELOVANJA KORELACIJE MERENJA SA DIGITALNIH UREĐAJA

U cilju analize uticaja korelacije merenja sa digitalnih uređaja, ovi uređaji su postavljeni na lokacijama u mreži označenim na slici 1. Lokacije digitalnih uređaja su izabrane proizvoljno, bez korišćenja tehnika optimalnog postavljanja mernih uređaja.



Slika 1. Lokacije digitalnih mernih uređaja

U delu 4.1 je pokazano da korelacija između merenja koje obezbeđuje jedan digitalni uređaj strogo zavisi od samih izmerenih veličina. Zbog toga se u ovom delu ne uzima specificirana vrednost faktora korelacije, već se proračun vrši uzimajući u obzir stvarno ponašanje digitalnih uređaja. Na početku su prisutna merenja napona i struje za koje se uzima da je varijansa u slučaju merenja modula $\sigma_V^2 = \sigma_I^2 = 10^{-4}$, a u slučaju merenja faznog ugla $\sigma_\varphi^2 = 3.364 \cdot 10^{-5}$.

U tabeli 1. prikazana je srednja vrednost RMSE za module napona i fazne uglove napona za sve čvorove u mreži sa i bez modelovanja korelacije u proračunima WLS estimacije stanja. Iz tabele se vidi da uključivanjem vandijagonalnih elemenata u matricu kovarijansi merenja nema uticaja na smanjenje srednje vrednosti RMSE za module napona čvorova. Smanjenje srednje vrednosti RMSE prisutno je samo za fazne uglove napona. Na osnovu rezultata može se zaključiti da je postignuto neznatno poboljšanje u tačnosti procene promenljivih stanja.

Tabela 1. Srednja vrednost RMSE za digitalne uređaje

Model	Srednja vrednost RMSE	
	modul napona [%]	fazni ugao napona [rad]
bez korelacije	0.4957	2.7714×10^{-3}
sa korelacijom	0.4957	1.8429×10^{-3}

6. REZULTATI MODELOVANJA KORELACIJE PSEUDO MERENJA

Za vrednost varijanse modula napona je uzeta vrednost $\sigma_V^2 = 10^{-4}$, za tok aktivne i reaktivne snage $\sigma_P^2 = \sigma_Q^2 = 9 \cdot 10^{-4}$, dok je za pseudo merenja vrednost varijanse $\sigma_{P_i}^2 = \sigma_{Q_i}^2 = 0.01$. Za faktor korelacije je uzeta vrednost od 0.8.

U tabeli 2. prikazana je srednja vrednost RMSE za modul napona i fazni ugao napona čvorova kada nije modelovana inter-node korelacija i kada jeste. Iz rezultata se može videti bitno poboljšanje u smanjenju srednje vrednosti RMSE i za module napona i za fazne uglove napona.

Tabela 2. Srednja vrednost RMSE za inter-node korelaciju

Model	Srednja vrednost RMSE	
	modul napona [%]	fazni ugao napona [rad]
bez korelacije	0.1371	1.2286×10^{-3}
sa korelacijom	0.0279	2.64×10^{-5}

U drugom delu je urađeno modelovanje intra-node korelacije. Rezultati su prikazani u tabeli 3. Uzimajući u obzir korelaciju između injektiranja aktivne i reaktivne snage istog čvora iz tabele 3 vidi se uticaj prvenstveno na estimaciju faznog ugla napona. Zapravo, uzimajući u obzir intra-node korelaciju pruža se dodatna informacija o faktoru snage injektiranih snaga čvorova.

Tabela 3. Srednja vrednost RMSE za intra-node korelaciju

Model	Srednja vrednost RMSE	
	modul napona [%]	fazni ugao napona [rad]
bez korelacije	0.1371	1.2286×10^{-3}
sa korelacijom	0.1014	0.6571×10^{-3}

Test kojim su uzeti u obzir i inter-node korelacija i intra-node korelacija je takođe izveden. U tabeli 4 su prikazani rezultati srednje vrednosti RMSE za ovaj test. Na osnovu srednje vrednosti RMSE uočava se kumulativni uticaj korelacija koje su izvedene u prethodnim testovima.

Tabela 4. Srednja vrednost RMSE za inter-node i intra-node korelaciju

Model	Srednja vrednost RMSE	
	modul napona [%]	fazni ugao napona [rad]
bez korelacije	0.1371	1.2286×10^{-3}
sa korelacijom	0.0293	0.3929×10^{-3}

U cilju daljeg istraživanja uticaja korelacije na estimirane vrednosti urađena je analiza u kojoj su menjane vrednosti faktora korelacije. Vrednosti faktora korelacije su smanjene za 25%, 50% i 75% u odnosu na referentnu vrednost faktora korelacije za koju je uzeta vrednost od 0.8. Rezultati su prikazani u tabeli 5. Vidi se da je veći uticaj postignut sa većim faktorom korelacije, odnosno da su srednje vrednosti RMSE i za modul napona i za fazni ugao napona manje sa korišćenjem većeg faktora korelacije. Na kraju je potrebno primetiti da je uticaj koji je postignut promenom faktora korelacije nelinearan zbog postojanja i indirektnih unakrsnih faktora korelacije.

Tabela 5. Srednja vrednost RMSE za različite vrednosti faktora korelacije

faktor korelacije	Srednja vrednost RMSE	
	modul napona [%]	fazni ugao napona [rad]
0.8	0.0293	0.3929×10^{-3}
0.6	0.0586	0.7643×10^{-3}
0.4	0.0950	0.9857×10^{-3}
0.2	0.1171	1.0786×10^{-3}

7. ZAKLJUČAK

U radu je prikazano modelovanje korelacije merenja sa digitalnih uređaja i modelovanje različitih varijanti korelacije pseudo merenja u proračunu statičke estimacije stanja na primeru test mreže sa 14 čvorova. Za proračun je upotrebljen najčešće korišćeni algoritam u statičkoj estimaciji stanja, a to je WLS metod.

Različiti tipovi korelacije merenja su analizirani. Za digitalne merne uređaje utvrđeno je neznatno poboljšanje u tačnosti procene promenljivih stanja uključivanjem korelacije merenja.

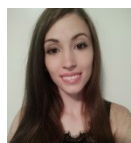
Analizirane su dve vrste korelacije pseudo merenja: inter-node i intra-node. Zaključeno je da u ovom slučaju, pravilnim uključivanjem korelacije pseudo merenja u matricu kovarijansi merenja vodi ka značajnijem poboljšanju u tačnosti procene promenljivih stanja.

Na kraju, testovi pokazuju i da je uticaj modelovanja korelacije pseudo merenja na tačnost procene promenljivih stanja veći pri korišćenju većeg faktora korelacije u proračunima. Međutim, usled postojanja indirektnih unakrsnih faktora korelacije pri uključivanju i inter-node i intra-node korelacije, uticaj promene faktora korelacije je nelinearan.

8. LITERATURA

- [1] A. Monticelli, *State estimation in electric power systems, a generalized approach*, Kluwer Academic Publishers, Norwell, 1999.
- [2] Y. Chen: Weighted-Least-Square (WLS) State Estimation, *PNNL*, decembar 2015.
- [3] E. Caro, A. J. Conejo, R. Mingez: Power System State Estimation Considering Measurement Dependencies, *IEEE*, novembar 2009.
- [4] C. Muscas, M. Pau, P. A. Pegoraro, S. Sulis: Effects of Measurements and Pseudo-measurements Correlation in Distribution System State Estimation, *IEEE*, 2012.
- [5] V. Krsman: *Specijalizovani algoritmi za detekciju, identifikaciju i estimaciju loših podataka u elektrodistributivnim mrežama*, doktorska disertacija, Fakultet tehničkih nauka, Novi Sad, 2014.
- [6] C. Muscas, M. Pau, P. A. Pegoraro, S. Sulis: Impact of Input Data Correlation on Distribution System State Estimation, *IEEE*, 2013.

Kratka biografija:



Milena Lisov rođena je u Novom Sadu 1994. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnika i računarstvo – Elektroenergetski sistemi odbranila je 2017. god. Iste godine upisala je master studije.

NIGHTMARE MAGIC – ПРИМЕР УПОТРЕБЕ ЕЛЕКТРОЕНЦЕФАЛОГРАФИЈЕ И EMOTIV EPOC УРЕЂАЈА У РАЧУНАРСКИМ ИГРАМА

NIGHTMARE MAGIC – AN EXAMPLE OF USING ELECTROENCEPHALOGRAPHY AND EMOTIV EPOC HEADSET IN COMPUTER GAMES

Марина Радовановић, Факултет техничких наука, Нови Сад

Област – ПРИМЕЊЕНЕ РАЧУНАРСКЕ НАУКЕ И ИНФОРМАТИКА

Кратак садржај – Електроенцефалографија (EEG, Electroencephalography) препознаје мерљиву електричну активност унутар мозга која се јавља при активирању неурона. За запис ових сигнала сензори се постављају преко скалпа, а један од BCI (Brain-Computer Interface, интерфејс мозак-рачунар) уређаја који користи ову неинвазивну методу је Emotiv EPOC, који омогућава управљање апликацијама путем мисли или израза лица. Циљ овог рада је истраживање практичне примене конкретног BCI уређаја у прављењу једноставне игре. Мерењем EEG таласа и детекцијом израза лица корисника управља се бацањем магије унутар игре Nightmare Magic, пројекта који је замишљен као демо за упознавање са радом Emotiv EPOC уређаја и почетна тачка за будућа истраживања.

Кључне речи: Електроенцефалографија, интерфејс мозак-рачунар, рачунарска игра.

Abstract – Electroencephalography (EEG) recognizes the measurable electrical activity inside the brain which appears due to neural activity. In order to capture these signals sensors are placed over the scalp, and one BCI (Brain-Computer Interface) headset that uses this non-invasive technique is Emotiv EPOC, which enables application control by using thoughts or facial expressions. The aim of this paper is researching the practical application of a concrete BCI device for making a simple computer game. By measuring EEG waves and detecting facial expressions of the user magic is cast in the game Nightmare Magic, a project intended as a demo for familiarization with how Emotiv EPOC works and a starting point for future research.

Keywords: Electroencephalography, brain-computer interface, computer game.

1. УВОД

BCI (Brain-computer interface, интерфејс мозак-рачунар) игре су рачунарске игре које користе мождане таласе за управљање игром. BCI уређаји омогућавају играње рачунарских игара и од стране особа са посебним потребама, док BCI рачунарске

игре могу бити доступније и интересантније од неких врста терапија или вежби за рехабилитацију. Електроенцефалографија (electroencephalography, EEG) је BCI метода која омогућава препознавање сигнала који се јављају при активацији можданих неурона, при чему се сензори постављају преко скалпа.

Централни принцип BCI система је осматрање мождане активности корисника и способност разликовања различитих образаца мождане активности, где се сваки повезује са одређеном намером или менталним задатком [1]. Слика 1 приказује уопштenu архитектуру EEG-BCI система. Мождана активност се снима помоћу преносног уређаја. Сирови сигнали се првобитно процесирају и трансформишу како би се издвајале неке значајне одлике.



Слика 1. Уопштена архитектура BCI система за управљање уређајима попут роботске руке или моторизованих инвалидских колица.

Издвајање образаца обухвата раздвајање корисних EEG података од шума и њихово поједностављивање, како би се могла извршити класификација тј. одлучивање шта издвојени подаци представљају. Не постоји идеалан начин за издвајање образаца из EEG података. Неке од метода које користе савремени BCI уређаји су: Хјортови параметри (начин описивања нормализованог нагиба података), вејвлет трансформација (wavelet transform), Фуријеова трансформација, као и различите врсте филтера. Основни обрасци од кога зависе EEG-BCI системи су потенцијали везани за догађај (ERP, event-related potential) и промене везане за догађај у специфичним опсезима фреквенције.

Издвојени обрасци се затим прослеђују ка неком математичком моделу (статички класификатори или неуронске мреже). Након процеса тренинга, овај модел рачуна одговарајуће менталне команде за управљање уређајем. Не постоји стандардизован

НАПОМЕНА:

Овај рад је проистекао из мастер рада чији ментор је био др Драган Дину, доцент.

начин класификације издвојених података *BCI* система. Да би се улазни подаци мапирани на познате категорије *EEG* архетипова, користе се различита средства попут неуронских мрежа, прагова и препознавања образаца [2].

Конечно, визуелна повратна информација, или на пример тактилна стимулација, обавештавају корисника о перформанси уређаја како би научио одговарајуће стратегије за ментално управљање и направио промене како би извршио задатак. Повратна информација у *BCI* системима је од изузетне важности јер омогућава кориснику да разуме какве мождане таласе је управо произвео, и да научи које понашање се ефективно може класификовати и контролисати. Повратна информација може бити у визуелном или аудиторном облику, или чак хаптичка сензација.

2. ПРЕГЛЕД СТАЊА У ОБЛАСТИ

Hans Berger, немачки неуропсихијатар, је први описао постојање људских *EEG* сигнала 1920-их година а свој први извештај о снимку *EEG* активности људског мозга у трајању од три минута је објавио 1929. године [3]. За време 1930-их година, *Berger* је снимио први *EEG* човековог спавања, осматрајући вретенасте структуре. Након тога, наставио је да изучава људске *EEG* обрасце при операцијама мозга, епилепсији, можданим дисфункцијама и менталним активностима.

Иако у то време термин *BCI* још увек није установљен, један од најранијих примера функционалног интерфејса мозак-машина је био комад *Music for Solo Performer* (1965), који је компоновао *Alvin Lucier*. Ова композиција користи *EEG* и хардвер за процесирање аналогних сигнала (филтере, појачаваче и плочу за миксовање) како би се симулирали акустични ударачки инструменти. За извођење композиције, потребно је произвести алфа таласе и на тај начин „свирати“ различите инструменте путем звучника који су постављени у близини или директно на самим инструментима [4].

Иако је идеја о коришћењу *EEG* таласа као улазних сигнала за *BCI* системе постојала дуже време, тек недавно су развијени прави *BCI* уређаји који раде базирајући се на *EEG* улазном сигналу [5]. До 2000. године, истраживачи су конструисали уређај за превођење мисли, за потпуно парализоване пацијенте. Уређај је омогућавао пацијентима да одаберу слово на основу својих мисли, али је процес одабира слова био дуготрајан и не потпуно прецизан [6]. До 2008. године, истраживачи из Швајцарске, Белгије и Шпаније су произвели асинхрони *BCI* систем који је управљао моторизованим инвалидским колицима са високим степеном прецизности, али је и даље имао одређене недостатке [7].

Данас, 2010 *DARPA* буџет издваја 4 милиона долара за развој програма базираног на *EEG* технологији, под називом *Silent Talk*. Овај програм би требао да омогући комуникацију корисник-корисник на бојном пољу без коришћења говора путем анализе неуралних сигнала [8].

3. МЕТОДОЛОГИЈА

Како би се извршило тестирање развијена је апликација *Nightmare Magic*, рачунарска игра којом се може управљати коришћењем *Emotiv EPOC* уређаја. Играч баца магију променом израза лица, док се ка њему крећу утваре.

У наставку овог поглавља биће описан принцип рада неинвазивне *BCI* методе електроенцефалографије и начин рада са уређајем *Emotiv EPOC*. Након тога биће анализиран пример примене конкретног *BCI* уређаја при управљању једноставном игром развијаном у *Unity engine*-у.

3.1. Електроенцефалографија (EEG)

Неинвазивна *BCI* метода, електроенцефалографија, се заснива на томе да активност неурона у мозгу узрокује стварање *EEG* таласа који се могу измерити коришћењем електрода осетљивих на промену постсинаптичких потенцијала неурона у церебралном кортексу. Електроде се постављају на скалп у складу са стандардом за мерење различитих регија мозга (*10-20 International System of Electrode Placement*) [9]. Свака електрода има стандардну осетљивост од $7\mu V/mm$ и користи просечну вредност потенцијала измерених у близини сензора. Добијене просечне вредности се затим појачавају и комбинују како би се добијена ритмичка активност могла класификовати према фреквенцији (табела 1) [10].

Опсег	Фреквенција (Hz)
Delta	1 – 4
Theta	4 – 7
Alpha	7 – 13
Beta	13 – 30
Gamma	30 +

Табела 1. *EEG* опсези и фреквенције.

Истраживања која изучавају *EEG* технологију се базирају на чињеници да посматрана ритмичка активност зависи од менталног стања и да на њу могу да утичу ниво свести или различите менталне болести. Често се пореде *EEG* таласи код будних и спаних пацијената, а посматрају се и различити маркери код неуобичајених *EEG* таласа који могу да укажу на болести као што су епилепсија или Алцхајмер.

Једна од тешкоћа при коришћењу *EEG* мерења је непоузданост података услед артефаката као што су електрични сигнали који не потичу од кортикалних неурона а детектовани су од стране сензора. Неки од најчешћих узрока артефаката су померање очију и трептање, као и коришћење мишића скалпа или врата, или пак лош контакт између скалпа и електрода [11].

3.2. *Emotiv EPOC* уређај

Emotiv EPOC уређај напаја допуњива литијумска батерија од 12Ah и чији је ефективни опсег фреквенција од 0.16Hz до 43Hz. Уређај садржи 14 *EEG* сензора, додатне *CMS* (*Common Mode Sense*) и *DRL* (*Driven Right Leg*) референце за формирање

повратне петље и референцирање осталих мерења, и жироскоп.

Један од софтверских пакета који се могу користити при развоју апликација је *Emotiv SDK Community Edition*, који омогућава развој софтвера компатибилног са *Emotiv* уређајима. *Emotiv API* је доступан као *ANSI C* интерфејс имплементиран у библиотеци *edk.dll* [12], где *Emotiv EmoEngine* класа представља логичку апстракцију функционалности. *EmoEngine* комуницира са *Emotiv* уређајем, прима препроцесиране *EEG* податке и податке жироскопа, чува подешавања специфична за корисника или апликацију, врши постпроцесирање и преводи резултате *Emotiv* детекције у структуру која се назива *EmoState* [12]. За одзив у скоро реалном времену, већина апликација би требала да потражује нове *EmoState* структуре барем 10-15 пута у секунди, што се обично дешава у главној петљи апликације или, у случају рачунарских игара, при периодичном упиту улазних уређаја.

Могуће је препознавање 12 израза лица: трептај, миг десним или левим оком, поглед десно или лево, подигнуте обрве, намрштене обрве, осмех, стиснути зуби, десни или леви полуосмех и смејање. За успешну детекцију израза лица потребно је најпре извршити тренинг за неутралан израз лица, а затим и за остале изразе лица који ће се препознавати.

Детекција менталних команди прати и интерпретира свесне мисли корисника. Разликују се 13 активних мисли: гурати, вући, подићи, спустити, лево, десно, ротирати лево, ротирати десно, ротирати у смеру казаљке на сату, ротирати у смеру супротном од смера казаљке на сату, ротирати ка напред, ротирати ка назад и нестајање.

4. СОФТВЕРСКА ИМПЛЕМЕНТАЦИЈА И РЕЗУЛТАТИ

Игра *Nightmare Magic* нема конкретан крај и циљ, јер је замишљена као опуштајуће (*casual*) искуство, са фокусом на мапирање израза лица са врстом магије која се баца и упознавање са радом *Emotiv EPOC* уређаја. Лик којим се управља у игри сања и налази се у замишљеном свету окружен утварама. Постоје три врсте утвара, при чему боје њихових огртача указују на врсту магије која их може уништити, а контроле утичу на врсту и правац бацања магије.

Пре него што корисник започне бацање магије унутар игре коришћењем *Emotiv EPOC* уређаја, потребно је извршити калибрацију уређаја комплетирањем неопходног тренинга. Калибрација осигурава да ће се улазни сигнали карактеристични за интеракцију са уређајем од стране специфичног корисника интерпретирати на прави начин и што је могуће прецизније. *Emotiv* интерфејс може функционисати унутар апликације и без извршеног тренинга, користећи уграђену метрику са просечним вредностима, али то обично доводи до нетачне детекције сигнала и неочекиваних резултата.

Потребно је извршити три тренинга како би се могли препознавати одговарајући управљачки сигнали унутар игре: неутрални израз лица, подигнуте обрве и стиснути зуби. При препознавању израза лица у

апликацијама које као улазни уређај користе *Emotiv EPOC*, неопходно је најпре извршити тренинг неутралног израза лица, који се користи као параметар за упоређивање свих осталих израза. Сваки израз који се користи као засебан управљачки сигнал мора имати засебан тренинг, поред обавезног тренинга за неутрални израз лица.

Сваки померај мишића лица при калибрацији се детектује од стране уређаја и може да доведе до непрецизног или чак потпуно погрешног препознавања управљачких сигнала, због чега је битно да корисник током читавог периода одржи израз лица што сличнији оном који ће користити за бацање конкретне магије. Ништа не спречава корисника да тренинг за подигнуте обрве изврши, на пример, са спуштеним обрвама и стиснутим зубима, али ће у том случају услед поређења са уграђеним просечним вредностима за тај израз лица доћи до велике разлике и управљање ће бити потпуно погрешно. Уколико се током извршења одређеног тренинга јављао шум или корисник мисли да би се тренинг требао поновити уз другачији израз лица или бољу концентрацију, могуће је уклонити тренинг а затим га извршити поново.

Извршење тренинга за *Emotiv EPOC* уређај се односи на калибрацију уређаја за тренутног корисника. Уколико се корисник промени, потребно је поново извршити све тренинге за новог корисника, како би се управљачки сигнали препознавали довољно прецизно и како би се обезбедило тачно управљање у складу са очекивањима.

За олакшану комуникацију са *EmoEngine* системом и смањење зависности других скрипти од *Emotiv API*-ја, написана је скрипта *EmotivConnectionController.cs*, која се користи као компонента на објекту унутар сцене у *Unity*. Ова компонента је активна од тренутка покретања до тренутка затварања апликације и пружа све потребне информације и догађаје у вези са повезаношћу *Emotiv EPOC* уређаја и апликације.

Интеракција са *EmoEngine*-ом је могућа путем скрипте *EmoEngine.cs* која је у директној повезаности са скриптом *EdkDll.cs*. Класа *EdkDll* представља *managed (C#)* верзију *EmoEngine*-а, који се у пројекат импортује као *edk.dll*.

Играч нема могућност кретања или промене позиције али може да нишани променом ротације, што се постиже помоћу жироскопа који пружа информације о ротацији *Emotiv* уређаја на глави корисника. Коришћење жироскопа унутар апликације не захтева посебан тренинг.

У игри су присутне три врсте утвара: црвене, плаве и црне. Црвене утваре се могу уништити бацањем ватрене магије, а отпорне су на електричну (шок) магију, док плаве утваре функционишу на супротан начин. Црне утваре се могу уништити било којом од две доступне магије. Бацање ватрене магије се иницира подизањем обрва док је бацање електричне магије могуће стискањем зуба.

Уколико је ново стање једно од стања која се очекују и на које је потребно одреаговати (подигнуте обрве или стиснути зуби), проверава се да ли је могуће бацити нову магију у односу на задати временски интервал који означава потребну паузу између два

бацања. При бацању магије користи се анимација видљивог оружја, инстанцира се објекат као визуелна репрезентације магије и репродукују звучни ефекти у складу са врстом магије која је бачена.

При удару у утвару која није отпорна на ватрену магију, изазива се ватрена експлозија, уз пратећи звучни ефекат који дочарава осећај сагоревања, док се при удару у утвару која није отпорна на електричну магију појављује визуализација струје, уз пратећи звучни ефекат који дочарава осећај струјног удара.

5. ЗАКЉУЧАК

При коришћењу *BCI* уређаја за генерисање управљачких сигнала, корисник добија посебан осећај моћи, попут суперхероја. Овај осећај изузетно доприноси искуству играња рачунарских игара, када корисник на кратко време напушта свакодневицу и прелази у други свет где може да се поистовети са ликом по избору. Бацање магије не даје исти осећај када се дешава на притисак тастера и када се изазове мишљењем о магији, или променом израза лица.

Један од аспеката који спречавају широку употребу *BCI* уређаја као улазних уређаја у рачунарским играма је чињеница да је пре сваког коришћења ради препознавања управљачких сигнала потребно извршити калибрацију односно кориснички тренинг. Овај процес се понавља при свакој промени корисника, како би препознавање образаца мишљења или израза лица било прецизније, при чему са порастом броја различитих управљачких сигнала расте и време потребно за тренинг. Такође, након одређеног времена коришћења уређаја потребно је размонтирати сензоре како би се поново наковасили раствором соли и побољшао квалитет сигнала при контакту са површином главе, што захтева искључивање уређаја и додатно време.

При управљању је изузетно важно да је претходно извршен тренинг тј. да је уређај калибрисан за тренутног корисника, како би улазни сигнали били прецизно интерпретирани. Уколико није извршен тренинг, при детекцији израза лица и најблажи покрет мишића лица може да изазове шум који ће бити погрешно интерпретиран (бацање магије док играч држи израз лица који није од конкретног значаја за управљање).

Вишеплатформска природа *Unity Engine*-а омогућава лако прилагођавање апликација за различите улазне уређаје или систем на коме ће се апликација покретати. Процес развоја саме игре у смислу сцене, функционалност објеката који се инстанцирају или са којима се интерагује, анимација, кориснички интерфејс, интеграција звука, као и многи други аспекти уобичајених рачунарских игара не зависе од платформе за коју се апликација развија, што отвара многе могућности за развој.

Пристапачан *Emotiv API* чини писање кода и интеграцију *Emotiv EPOC* уређаја са *game engine*-ом као што је *Unity* прилично једноставно, чиме смањује потребан напор за развој игара и апликација којима се може управљати помоћу овог уређаја. Иако се *BCI* уређаји још увек не користе често као улазни уређаји за управљање рачунарским играма, њихова популарност расте заједно са бројем потенцијалних примена у индустрији.

6. ЛИТЕРАТУРА

- [1] José del R. Millán, Pierre W. Ferrez, Ferran Galán, Eileen Lew, Ricardo Chavarriaga, 2007, *Non-Invasive Brain-Machine Interaction*, International Journal of Pattern Recognition and Artificial Intelligence Vol. XX, No. X, 1–13.
- [2] Adlakha Amit, 2002, *Single Trial EEG Classification*. Swiss Federal Institute of Technology.
- [3] Aatif M. Husain, Saurabh R. Sinha, *Continuous EEG Monitoring: Principles and Practice*, pp. 6.
- [4] Volker Straebel, Wilm Thoben, 2014, *Alvin Lucier's music for solo performer: experimental music beyond sonification*, Organised Sound, 19 (1): 17–29.
- [5] Vidal J., 1973, *Toward direct brain-computer communication*, Annual Review of Biophysics and Bioengineering, 2 (1): 157–80.
- [6] Birbaumer N. et al., 2000, *The Thought Translation Device (TTD) for Completely Paralyzed Patients*, IEEE Transactions on Rehabilitation Engineering, Volume 8, No. 2, pp. 190-193.
- [7] Galán F. et al, 2008, *A Brain-Actuated Wheelchair: Asynchronous and Non-invasive Brain-Computer Interfaces for Continuous Control of Robots*, Clinical Neurophysiology, Volume 119, Issue 9, pp. 2159-2169.
- [8] Drummond K., 2009, *Pentagon Preps Soldier Telepathy Push*, Wired Magazine.
- [9] Niedermeyer, Ernst and da Silva, Fernando Lopes, 2005, *Electroencephalography: Basic Principles, Clinical Applications and Related Fields, Fifth Edition*, Lippincott Williams & Wilkins, pp 140.
- [10] Nunez PL., Srinivasan R., 1981, *Electric Fields of the Brain: The Neurophysics of EEG*, Oxford University Press.
- [11] Rowan A. James, 2003, *Primer of EEG*, Elsevier Science, Philadelphia, PA.
- [12] Emotiv getting started guide, version 3.5.0., EMOTIV Inc 2017.

Кратка биографија:



Марина Радовановић рођена је у Новом Саду 1993. год. Мастер рад на Факултету техничких наука из области Примењене рачунарске науке и информатика – Мултимедија и рачунарске игре одбранила је 2019. год.

OBJEKTNA PARADIGMA RUBY PROGRAMSKOG JEZIKA SA KOMPARATIVNOM ANALIZOM RUBY ON RAILS I JAVA SPRING OKRUŽENJA**THE OBJECTIVE PARADIGM OF THE RUBY PROGRAMMING LANGUAGE WITH COMPARATIVE ANALYSIS OF RUBY ON RAILS AND JAVA SPRING ENVIRONMENT**Olja Anđelovski, Srđan Popov, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – U radu je analizirana objektno-orijentisana paradigma Ruby programskog jezika uz komparativnu analizu Ruby on Rails i Java Spring okruženja za kreiranje web aplikacija. Implementirana aplikacija Travel je standardna web aplikacija za upravljanje šifarnicima, pregled, kreiranje i rezervisanje smeštaja.

Ključne reči: Ruby, Objektno-orijentisana paradigma, Ruby on Rails, Java, Spring

Abstract – The paper analyzes the object-oriented paradigm of the Ruby programming language with a comparative analysis of the Ruby on Rails and Java Spring environment for creating web applications. Implemented Travel application is a standard web application for managing codebooks, viewing, creating and booking accommodations.

Keywords: Ruby, Object-oriented paradigm, Ruby on Rails, Java, Spring

1. UVOD

Objektno-orijentisana paradigma se bavi modelovanjem entiteta koji se zovu objekti. Omogućava upotrebu funkcionalnosti bez poznavanja specifičnosti ponašanja ili podataka unutar objekata, dozvoljava definisanje apstraktnih tipova gde je funkcionalnost samo delimično definisana kao i mnoge druge pogodnosti.

Cilj rada je upoznavanje koncepata objektno-orijentisane paradigme i implementacija istih u okviru programskog jezika Ruby. Na studiji slučaja je izvršena komparativna analiza Ruby on Rails, kao jednog od najpopularnijih okruženja za razvoj klijent-server baziranih aplikacija u Java Spring okruženju.

2. POJMOVI I KONCEPTI OBJEKTNO-ORIJENTISANOG PROGRAMIRANJA U PROGRAMSKOM JEZIKU RUBY**2.1. Klase i objekti**

Osnovna uloga klase je kreiranje objekata, inicijalizacija početnih vrednosti atributa, kao i implementacija ponašanja u okviru metoda.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Srđan Popov, vanredni profesor.

Klase se definišu uz pomoć ključne reči **class** iza koje ide naziv klase koji uvek započinje velikim slovom, nakon toga se pišu atributi i metode i obavezno se završava sa **end**.

Objekat predstavlja komponentu koja sadrži attribute i metode i uz pomoć metoda može da izvršava određene zadatke i utiče na ostale objekte. Kreira se pozivanjem metode **new**, a potom se u pozadini poziva metoda **initialize** kojom se inicijalizuju vrednosti atributa objekata.

2.1.1. Klasne metode

Klasne metode se pozivaju nad samim klasama. Kreiraju se dodavanjem prefiksa **self** ispred naziva metode.

2.1.2. Pristupne metode

Pristupnim metodama pristupamo atributima unutar klase. Pristupne metode u Ruby programskom jeziku su: **attr**, **attr_reader**, **attr_writer** i **attr_accessor**.

2.2. Konstruktor

Konstruktor je metoda kojom se kreira objekat i vrši njegova inicijalizacija. Konstruktor se u Ruby programskom jeziku kreira definisanjem metode **initialize**, koja ima **private** pravo pristupa. Ukoliko postoji konstruktor, pozivom klasne metode **new** pozvaće se metoda **initialize** i kreiraće se inicijalizovani objekat, u suprotnom biće pozvan podrazumevani konstruktor i kreiraće se neinicijalizovan objekat.

2.3. Destraktor

U Ruby programskom jeziku brisanje objekata i oslobađanje memorije vrši sakupljač smeća. Programer ne upravlja memorijom, već Ruby inicira dodeljivanje memorije. Prilikom kreiranja objekta Ruby zahteva od kernela memoriju. Ukoliko nema slobodnih slotova sakupljač smeća proverava objekte koji više nisu referencirani od strane drugih objekata i oslobađa ih.

2.4. Moduli

Moduli predstavljaju skup konstanti, klasnih varijabli i metoda instanci. Modul se definiše uz pomoć ključne reči **module** iza koje ide ime modula sa velikim početnim slovom, nakon toga ide struktura i završava se sa **end**. Najvažnije uloge modula su: *kreiranje prostora imena* i *implementiranje mixin mehanizma*.

2.4.1. Prostori imena

Moduli nam daju mogućnost da definišemo prostor imena, izolovani skup metoda i konstanti gde ne mora da se vodi računa o načinu na koji će biti uključene u druge module ili klase.

2.4.2. Mixin mehanizam

Uključivanje modula u definiciju klase predstavlja *mixin* mehanizam. Uz pomoć njega je moguće implementirati višestruko nasleđivanje. Moduli se u klase uključuju korišćenjem ključne reči *include* ili *extend*.

2.5. Nasleđivanje

Nasleđivanje je veza između klasa koja podrazumeva preuzimanje sadržaja nekih klasa, odnosno klasa-predaka i na taj način, uz mogućnost modifikacije preuzetog sadržaja i dodavanjem novog dobija se klasa-potomak [1]. U Ruby programskom jeziku je moguće naslediti isključivo jednu klasu dodavanjem operatora < nakon imena potklase, a pre imena natkalase koju želimo da naledimo. Višestruko nasleđivanje se implementira uz pomoć *mixin* mehanizama na taj način što se moduli uključuju u definiciju drugog modula ili klase.

2.6. Polimorfizam

Polimorfizam je koncept programiranja koji se odnosi na sposobnost varijabli, funkcija ili objekata da prihvate više oblika. Polimorfizam pruža jedan interfejs entitetima različitog tipa. U Ruby-u se ostvaruje upotrebom: *dinamičkog tipiziranja, nasleđivanja i modula*.

2.7. Enkapsulacija

Enkapsulacija je pojam koji se odnosi na sakrivanje informacija. Koristi se kako bi se sakrila unutrašnja reprezentacija objekta i njegovih stanja od strane spoljašnosti. Podaci i funkcije za izvršavanje akcija nad podacima se grupišu u jedinstvenu jedinicu – klasu.

Osnovna svrha enkapsulacije je da obezbedi sigurnost podataka u klasi. Da bi podaci unutar klase bili sigurni i zaštićeni od spoljašnjeg uticaja, neophodno je da se atributi definišu sa *private* pravom pristupa koji se kombinuju sa *public* metodama koje omogućavaju pristup atributima i izvršavanje zadataka. Još jedan način za enkapsulaciju podataka u Ruby-u jeste mehanizam za ugnježdavanje klasa i modula. Prednosti korišćenja enkapsulacije su: *sakrivanje informacija, ponovna upotrebljivost koda i jednostavno testiranje*.

2.8. Modularnost

Modularnost se odnosi na koncept kreiranja više različitih, relativno nezavisnih modula koji se povezuju i kombinuju u cilju formiranja kompletnog sistema. Modul predstavlja zasebnu softversku komponentu, koja ima određenu funkciju. Modularnost se u Ruby programskom jeziku postiže na više načina.

Jedna od mogućnosti je definisanje klasa koje imaju privatne attribute i kojima je ograničeno pravo pristupa, klasa ne sadrži spoljašnje članove i predstavlja izolovanu celinu. Takođe modularnost se može postići korišćenjem modula, kao i definisanjem klasa koje sadrže ugnježdene klase, odnosno module.

Prednosti modularnog programiranja su: *skraćivanje vremena pisanja koda, ponovna upotrebljivost, olakšano dizajniranje programa, olakšana čitljivost programa, olakšano otklanjanje grešaka*.

2.9. Konverzija

Da bismo omogućili rad sa podacima koji dolaze iz spoljnih izvora kao što su tastatura, baza podataka, odgovor api-a neophodno je izvršiti konverziju. U Ruby programskom jeziku je moguće izvršiti *implicitnu i eksplicitnu* konverziju.

Eksplicitna konverzija se vrši na izričit način, pozivanjem metoda za konverziju nad podacima koje želimo da konvertujemo u ciljani tip. Neke od metoda su: *to_s, to_i, to_a*, itd.

Implicitna konverzija se vrši prilikom izračunavanja izraza koji sadrže operande različitog tipa i Ruby je vrši automatski. Konverzija se vrši na taj način što se manje složeni operand konvertuje u složeniji. Metode koje treba implementirati da bi se obavila implicitna konverzija u slučaju da nije podržana su: *to_str, to_int, to_ary*, itd.

2.10. Preklapanje operatora

Preklapanje operatora predstavlja pridruživanje istog simbola različitim rutinama. Dva ili više operatora su preklopljena ukoliko za isti simbol imaju pridružen različit kod [2].

U programskom jeziku Ruby preklapanje operatora se vrši na taj način što se u korisnički definisanim tipovima definišu operatori sa posebnom logikom koja je prilagođena prosleđenim argumentima, odnosno njihovom tipu. Operatori u Ruby programskom jeziku predstavljaju poziv odgovarajućih metoda u pozadini.

2.11. Dinamičko povezivanje

Povezivanje predstavlja vezu između poziva metode i koda koji se izvršava. Postoje dva tipa povezivanja: *statičko i dinamičko*.

Statičko povezivanje kompajler može da reši u toku kompajliranja, dok se kod dinamičkog povezivanja na osnovu tipa objekta nad kojim se poziva metoda odlučuje koja će metoda izvršiti. Ruby podržava samo dinamičko povezivanje.

3. STUDIJA SLUČAJA

Za studiju slučaja je odabrana implementacija web aplikacije *Travel* koja je napisana u programskom jeziku Ruby u okviru Ruby on Rails okruženja, koristeći objektno-orijentisanu paradigmu. Izvršeno je poređenje sa aplikacijom napisanom u Javi koristeći Spring okruženje.

3.1. Specifikacija projekta

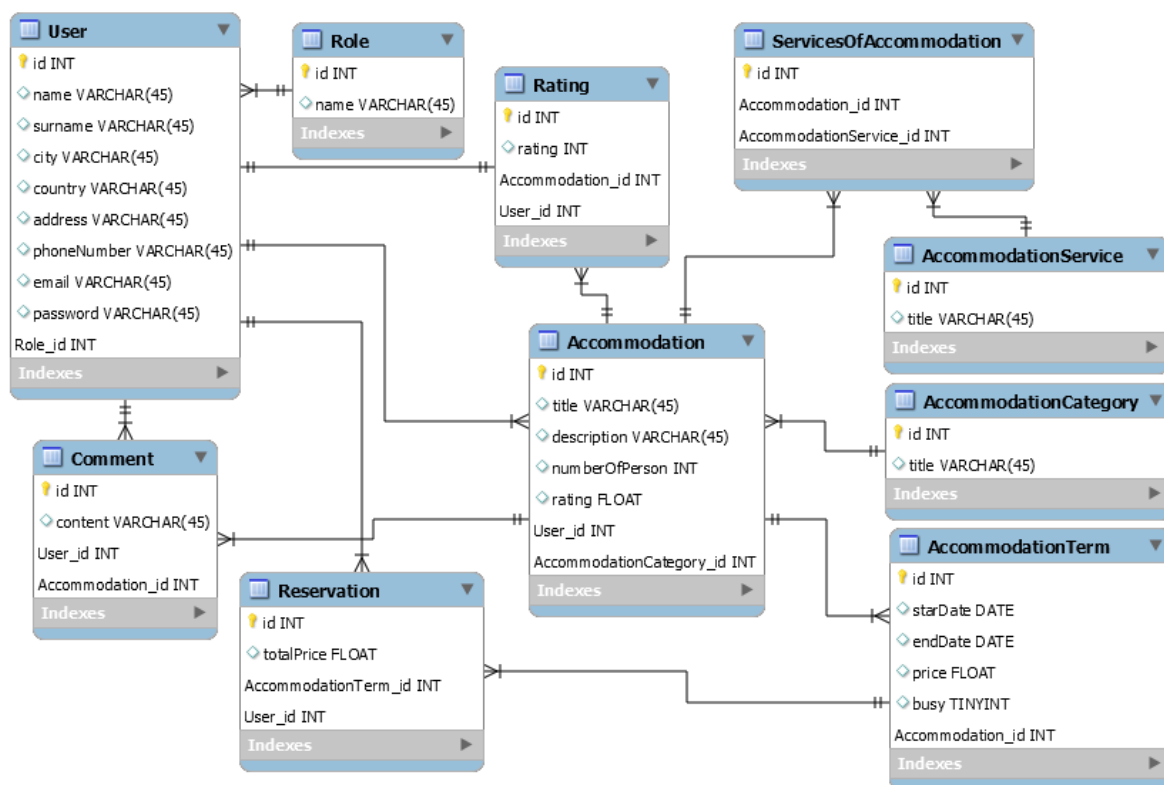
Aplikacija *Travel* je web aplikacija čija je osnovna funkcionalnost pregled i rezervisanje smeštaja, kao i obavljanje osnovnih crud operacija nad definisanim entitetima.

Za izvršavanje glavnih funkcionalnosti u okviru aplikacije su identifikovani sledeći korisnici: *administrator, agent i kupac*.

Administrator predstavlja predefinisano korisnika koji ima pristup bazi podataka i čiji je glavni zadatak dinamičko upravljanje entitetima u vidu brisanja, dodavanja i izmene kategorija smeštaja, servisa koje pruža smeštaj, kao i upravljanje registrovanim korisnicima.

Agent je zadužen za kreiranje smeštaja, kao i dodavanje termina u okviru smeštaja. Ima mogućnost brisanja i izmene smeštaja, kao i pregled rezervisanih smeštaja.

Kupac predstavlja krajnjeg korisnika, koji može da pregleda i rezerviše termine u okviru smeštaja, nakon rezervisanja ima mogućnost ocenjivanja i ostavljanja komentara za smeštaj u kome je boravio. Na slici 1. je prikazan model podataka.



Slika 1. Model podataka

3.2. Ruby on Rails

Ruby on Rails je jedan od najpopularnijih okruženja za razvoj klijent-server baziranih web aplikacija napisan u Ruby programskom jeziku. Rails koristi *MVC (Model-View-Controller)* arhitekturu čija je osnovna ideja ponovna upotreba postojećeg koda, olakšavanje razvoja koda, kao i razdvajanje koda na odvojene komponente. Pored MVC-a Rails podržava i ostale poznate šablone i paradigme, sa akcentom na principe *DRY (Don't Repeat Yourself)* i *COC (Convention Over Configuration)*. Korišćenjem konvencija prilikom kreiranja aplikacija, uvođenjem migracija i mnogih drugih koncepata Rails je imao veliki uticaj na razvoj web aplikacija.

3.3. MySQL

Za konfigurisanje MySQL baze podataka bilo je potrebno izmeniti konfiguracije u okviru *database.yml* datoteke koja je smeštena unutar *config* direktorijuma i instalirati *mysql2 adapter*.

U poređenju sa aplikacijom napisanom u Java programskom jeziku, koristeći Spring okruženje, u okviru koje se koristi H2 baza podataka možemo da primetimo da je jedina razlika u konfiguraciji u tome što prilikom dodavanja novih zavisnosti Spring sam vrši instalaciju.

3.4. Model

Za kreiranje prethodno definisanog modela u okviru Ruby on Rails platforme je korišćen *ActiveRecord* koji predstavlja model u MVC arhitekturi i koji je zadužen za predstavljanje poslovnih podataka i logike. Veze između entiteta se definišu asocijacijama.

Kreiranje modela i mapiranje u Spring-u, za razliku od Rails-a predstavlja teži posao i zahteva dobro poznavanje anotacija i njihovih mogućnosti. U Spring-u je neophodno ručno kreirati svaku od klasa, definisati attribute, konstruktore i metode koje dodatno treba anotirati kako bi se

uspešno izvršila konfiguracija. Model u Rails-u je dosta pregledniji, veliki deo koda se automatski generiše i ukoliko se ispoštuju konvencije prilikom imenovanja Rails oslobađa programera od dodatnog posla.

3.5. Controller

Za implementaciju sloja poslovne logike u Ruby on Rails-u je korišćen *ActionController* koji predstavlja kontroler u MVC arhitekturi. Kontroler je zadužen za proizvodnju odgovarajućeg izlaza nakon što ruter utvrdi koji od kontrolera treba da obradi zahtev.

Implementacija poslovne logike u Spring-u zahteva kreiranje klasa koje vrše uloge kontrolera i servisa.

Anotiranjem smo morali da izvršimo konfiguraciju ponasob za svaki od kontrolera i servisa. Kontroler kao i u Rails-u sadrži metode koje moraju da se anotiraju vrstom zahteva koji se mapira na datu metodu. Za razliku od Rails-a koji unutar metoda kontrolera ima implementiranu funkcionalnost, Spring odvaja zahtev od implementacije korišćenjem servisa u okviru kojih se implementiraju metode koje komuniciraju sa bazom podataka i vraćaju rezultat kontroleru. Implementacija koda u Ruby on Rails-u je brža od implementacije u Spring-u, potencijalne greške se na samom početku uklanjaju ukoliko se ispoštuje konvencija imenovanja asocijacija, kontrolera i modela.

3.6. View

Rails sam generiše odgovarajući direktorijum koji treba da sadrži stranice koje će biti prikazane korisniku. Interfejs koji je kreiran kako bi korisnik mogao da manipuliše web aplikacijom je napisan koristeći jQuery u okviru html stranica. Za kreiranje interfejsa je korišćeno i Bootstrap okruženje.

U aplikaciji napisanoj u Spring-u smo za korisnički interfejs koristili AngularJS. Mapiranje rezultata dobijenih od

strane kontrolera je jednostavnije u Rails-u. Metode ne sadrže povratne vrednosti već se vrši direktno mapiranje podataka sa stranice koja poziva kontroler na kontroler i obrnuto.

3.7. Autentifikacija

Autentifikacija i autorizacija predstavljaju ključne pojmove koje je neophodno implementirati tako da se postigne što veća bezbednost prilikom korišćenja web aplikacija koje sadrže osetljive podatke. U aplikaciji Travel u Rails-u je odabran *Devise* kao jedan od brzih i fleksibilnih rešenja za autentifikaciju. Devise nudi mehanizme heširanja lozinki, sadrži već kreirane forme za registraciju i logovanje, čuva i briše korisničke sesije itd. Pored sloja na serveru, da bi omogućili autentifikaciju potrebno je definisati odgovarajuće forme na klijentskoj strani. Za razliku od Devise-a koji u pozadini vodi računa o heširanju i čuvanju lozinki, u Spring-u sve stavke moraju da se implementiraju ručno u kodu uz pomoć *Spring Security* modula.

4. ZAKLJUČAK

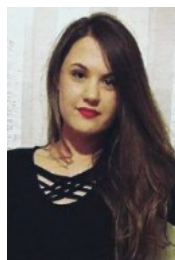
U radu je pokazano da se u Ruby programskom jeziku mogu u potpunosti implementirati koncepti objektno-orijentisane paradigme. Ruby nudi brojne mogućnosti u vidu metoda koje olakšavaju i ubrzavaju pisanje koda. Mnogi koraci u implementaciji koncepata se već nalaze kao podrazumevani u okviru programskog jezika.

Ruby on Rails okruženje omogućava kreiranje web aplikacija u Ruby programskom jeziku. Rails okruženje prati trendove u razvoju web aplikacija i kao takav zbog brzog razvoja aplikacija, konvencija koje pruža okruženje uživa popularnost. Posmatrano sa aspekta programera koji se prvi put susreće sa konceptima objektno-orijentisanog programiranja i web programiranja, Spring ima prednost zbog sličnosti sa standardnim Java aplikacijama. Sa aspekta programera koji ima iskustva u pisanju web aplikacija, Ruby on Rails je okruženje koje predstavlja moćan alat za brz razvoj aplikacija i uz puno fleksibilnosti predstavlja pravi izbor.

5. LITERATURA

- [1] Malbaški Dušan, Kupusinac Aleksandar, „Praktikum za vežbe iz objektno orijentisanog programiranja”, Novi Sad, Fakultet tehničkih nauka u Novom Sadu, 2010.
- [2] Malbaški Dušan, „Objektno orijentisano programiranje programski jezik java”, Novi Sad – Sremska Kamenica, Univerzitet Educons – Fakultet informacionih tehnologija, 2015.

Kratka biografija:



Olja Andelovski, rođena je 03.11.1995. u Beogradu. Završila je Osnovnu školu „Sava Žebeljan” u Crepaji. Nakon završene Osnovne škole upisuje srednju školu, Gimnaziju „Mihajlo Pupin” u Kovačici. 2014. godine upisuje „Fakultet tehničkih nauka” u Novom Sadu, smer „Računarstvo i automatika”. 2016/17. godine upisuje smer „Primenjene računarske nauke i informatika”. 2017/18. godine upisuje smer „Softversko inženjerstvo”. 2018. godine dobija zvanje Diplomirani inženjer elektrotehnike i računarstva. 2018/19. godine upisuje master akademske studije, smer „Primenjene računarske nauke i informatika – Elektronsko poslovanje”. 09.11.2018. godine dobija zvanje Saradnika u nastavi na Departmanu za računarstvo i automatiku na Fakultetu tehničkih nauka u Novom Sadu. kontakt: andjelovskiolja@gmail.com

U realizaciji Zbornika radova Fakulteta tehničkih nauka u toku 2018. godine učestvovali su sledeći recenzenti:

Aco Antić	Dušan Uzelac	Matija Stipić	Slobodan Dudić
Aleksandar Erdeljan	Duško Bekut	Milan Rapajić	Slobodan Krnjetin
Aleksandar Ristić	Đorđe Ćosić	Milan Simeunović	Slobodan Morača
Bato Kamberović	Đorđe Lađinović	Milan Trifković	Sonja Ristić
Biljana Njegovan	Đorđe Obradović	Milan Trivunić	Srđan Kolaković
Bogdan	Đorđe Vukelić	Milan Vidaković	Srđan Popov
Kuzmanović	Đula Fabian	Milena Krklješ	Srđan Vukmirović
Bojan Batinić	Đura Oros	Milica Kostreš	Staniša Dautović
Bojan Lalić	Đurđica Stojanović	Milica Miličić	Stevan Gostojić
Bojan Tepavčević	Filip Kulić	Mijodrag Milošević	Stevan Milisavljević
Bojana Beronja	Goran Sladić	Milovan Lazarević	Stevan Stankovski
Branislav Atlagić	Goran Švenda	Miodrag	Strahil Gušavac
Branislav Nerandžić	Gordana	Hadžistević	Svetlana Nikoličić
Branislav Veselinov	Milosavljević	Miodrag Zuković	Tanja Kočetov
Branislava Kostić	Gordana Ostojić	Mirjana	Tatjana Lončar -
Branislava	Igor Budak	Damnjanović	Turukalo
Novaković	Igor Dejanović	Mirjana Malešev	Toša Ninkov
Branka Nakomčić	Igor Karlović	Mirjana Radeka	Uroš Nedeljковиć
Branko Milosavljević	Ivan Beker	Mirko Borisov	Valentina Basarić
Branko Škorić	Ivana Katić	Miro Govedarica	Velimir Čongradec
Damir Đaković	Ivana Kovačić	Miroslav	Veran Vasić
Danijela Lalić	Ivana Miškeljin	Hajduković	Veselin Perović
Darko Čapko	Jasmina Dražić	Miroslav Popović	Vladimir Katić
Darko Marčetić	Jelena Atanacković	Mitar Jocanović	Vladimir Strezoski
Darko Reba	Jeličić	Mladen Kovačević	Vlado Delić
Dejan Ubavin	Jelena Borocki	Mladen Radišić	Vlastimir Radonjanin
Dejana Nedučin	Jelena Kiurski	Momčilo Kujačić	Vuk Bogdanović
Dragan Ivanović	Jelena Radonić	Nemanja	Zdravko Tešić
Dragan Ivetić	Jovan Petrović	Stanisavljević	Zoran Anišić
Dragan Jovanović	Jovanka Pantović	Nemanja Sremčev	Zoran Brujić
Dragan Kukolj	Ksenija Hiel	Nikola Đurić	Zoran Jeličić
Dragan Mrkšić	Laslo Nađ	Nikola	Zoran Mitrović
Dragan Pejić	Lazar Kovačević	Jorgovanović	Zoran Papić
Dragan Šešlija	Leposava Grubić	Nikola Radaković	Željen Trpovski
Dragana Bajić	Nešić	Ninoslav Zuber	Željko Jakšić
Dragana	Livija Cvetičanin	Ognjen Lužanin	
Konstantinović	Ljiljana Vukajlov	Pavel Kovač	
Dragana Šarac	Ljiljana Cvetković	Peđa Atanasković	
Dragana Štrbac	Ljubica Duđak	Petar Malešev	
Dragoljub Šević	Maja Turk Sekulić	Predrag Šiđanin	
Dubravka Bojanić	Marko Todorov	Radivoje Dinulović	
Dušan Dobromirov	Marko Vekić	Radovan Štulić	
Dušan Gvozdenac	Maša Bukurov	Slavica Mitrović	
Dušan Kovačević		Slavko Đurić	