УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

ЗБОРНИК РАДОВА
ФАКУЛТЕТА ТЕХНИЧКИХ НАУКА

Едиција: Техничке науке - зборници
Година: XXXVI
Број: 11/2021

Нови Сад

Едиција: „Техничке науке – Зборници“
Година: XXXVI Свеска: 11

Издавач: Факултет техничких наука Нови Сад
Главни и одговорни уредник: проф. др Срђан Колаковић, декан Факултета техничких Наука у Новом Саду

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
Срђан Колаковић. – 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 web 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 19.07.2021. до 29.09.2021. год., а који се промовишу 20.12.2021. год. То су оригинални прилози студената са главним резултатима њихових мастер радова.

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

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

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

- машинства,
- електротехнике и рачунарства и
- инжењерства информационих система.

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,
- мехатронике,
- математике у техници,
- заштите вода (TEMPUS) и
- управљања ризиком од катастрофалних догађаја и пожара.

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

Radovi iz oblasti: Elektrotehnika i računarstvo

STRANA

STRANA

Radovi iz oblasti: Inženjerstvo informacionih sistema

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 629
DOI: https://doi.org/10.24867/15AM01Nikacevic

TEHNIČKA EKSPLOATACIJA AUTOMATSKIH MENJAČA U MOTORNIM VOZILIMA

TECHNICAL EXPLOITATION OF AUTOMATIC TRANSMISSION IN MOTOR VEHICLES

Srđan Nikačević, Jovan Dorić, Marko Vilotić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj – *U radu je urađena analiza tehničke eksploatacije automatskih menjača u putničkim vozilima, njihovog održavanja i reparacije.*

Ključne reči: *Automobilska transmisija, automatski menjači, eksploatacija automatskih menjača, održavanje automatskih menjača*

Abstract – *This paper presents the analisis of automatic transmission systems in automotive aplications, their maintenance and repairing.*

Keywords: *Automotive transmission, automatic transmission, exploitation of automatic transmission systems, repairing of automatic transmission*

1. UVOD

Svakom vozilu, radnoj mašini ili pogonu, pogonjenim motorom SUS potrebna je neka vrsta redukdora, odnosno transmisije. Ova potreba dolazi iz radne karakteristike motora SUS, koja u osnovi nije prikladna onome čemu je namenjena, što je u slučaju vozila primarno stvaranje potrebne snage za pogon točkova i kretanje vozila.

Uloga transmisije jeste da pretvara moment i broj obrtaja motora u skladu sa potrebom na izlazu, u slučaju vozila to zavisi od samog vozila, puta, okoline i samog vozača. Različiti prenosnici snage razlikuju se po svojoj funkciji i svrsi, u putničkim vozilima se najčešće koriste višestepeni menjači u ulozi reduktora.

Shodno naretku tehnologije, kao i rastu potrebe za komforom u vožnji, na tržištu se sve više koriste tipovi menjača sa automatskom promenom stepena prenosa. Kako su s vremenom rasli zahtevi transmisije u vozilu pojavljivali su se različiti vidovi transmisije sa automatskom ili kontinualnom promenom stepena prenosa, kao što su AMT, DCT i CVT. Iako se među različitim regijama sveta razlikuju potražnje na tržištu transmisije, na svetskom nivou automatski menjači i dalje čine više od pola tržišta, ne uključujući manualne menjače [1].

Generalno gledano svaki proizvod da bi bio kompetetivan mora da bude odgovarajući za određenu namenu, da ima dobar odnos cene i kvaliteta, da bude lak za upotrebu i ne zagađuje okolinu. Kroz ove zahteve automatski menjači treba da budu konstruisani za prenos određenog obrtnog momenta i za upotrebu u određenim stilovima vožnje, da pozitivno utiču na potrošnju goriva i da budu kompetetivni po svojoj ceni [2].

Rani kvarovi ili prerana potreba za izmenom zamenskih delova automatskog menjača loše utiču na ekonomsku efikasnost kao i udobnost i samim tim smanjuju kompetetivnost menjača na tržištu. Životni vek B10 menjača u putničkim vozilima bi trebalo da bude veći od 150.000 pređenih kilometara [2]. Naravno preduslov da određeni menjač dostigne svoj živedni vek su propisana redovna održavanja i izmena zamenskih delova menjača.

Osnovni cilj rada je analiza eksploatacije automatskih menjača u praksi, sa fokusom na njihovom redovnom održavanju i reparaciji. Istraživanje je rađeno u praksi u radionici za popravku i održavanje automatskih menjača Garage021 u Novom Sadu.

2. KONSTRUKCIJA AUTOMATSKIH MENJAČA

U osnovi automatski menjač se sastoji iz hidrodinamičkog transformatora obrtnog momenta, planetarnih prenosnika i sistema za kontrolu promene stepena prenosa. Različite kombinacije prenosnih odnosa, različiti stepeni prenosa, postižu se preko HDTOM (hidrodinamički transformator obrtnog momenta) i različitih mogućnosti uparivanja zupčanika. Kontrola nad promenom stepena prenosa vrši se hidrauličkim putem, preko niza ventila, koji upravljaju različitim spojnicama i kočnicama time uparujući različite delove planetarnih zupčanika. Hidraulički sistem je kontrolisan od strane elektronskog sistema, putem ECU i niza senzora. Dva osnovna parametra za kontrolu automatskog menjača su broj obrtaja i opterećenje (praćeno preko položaja papučice gasa).

Konstrukcija i kontrola automatskog menjača mogu se podeliti u četiri osnovna elementa/sistema [1]:

1. Hidrodinamička transmisija. Postoje dva moguća rešenja: hidraulička spojnica i hidrodinamički transformator obrtnog momenta. Najčešće rešenje je HDTOM.
2. Mehanička transmisija. Sastoji se od planetarnih prenosnika i mehaničkih elemenata za promenu stepena prenosa (višelamelaste spojnice, pojasne kočnice i jednosmerna kvačila) koji vrše ulogu uparivanja različitih zupčanika.
3. Sistem hidrauličnog upravljanja. Sastoji se u osnovi od hidraulične pumpe, glavnog regulacionog ventila, prigušnih ventila i elektro-hidrauličkih proporcionalnih ventila. Za ulogu ima kontrolu nad menjačem, ali i podmazivanje i hlađenje elemenata menjača.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Boris Stojić, vanr. prof.

4. Sistem elektronskog upravljanja. Sastoji se iz tri celine, a to su: senzori, elektronska kontrolna jedinica (ECU) i aktuatori.

3. REDOVNO ODRŽAVANJE I SERVISI AUTOMATSKIH MENJAČA

Preporučeno vreme servisa razlikuje se u zavisnosti od proizvođača i uglavnom je referentni podatak broj pređenih kilometara vozila. Veliku razliku u učestalosti potrebnih servisa prave uslovi vožnje, kao i način vožnje automobila [1]. Opšte gledano servis automatskog menjača je potrebno raditi na otprilike 60.000 km, u slučaju većinski pogodnih uslova vožnje, koji podrazumevaju vožnju pretežno na autoputu i slično. Shodno težim uslovima vožnje, vreme potrebnog servisa se skraćuje. Ako se posmatra menjač na automobilu koji radi isključivo u teškim uslovima vožnje, kao što su na primer taksi vozila, preporuka je da se ovaj period duplo smanji na oko 30.000 km. U slučaju prosečnih gradskih vozača, preporuka je da se osnovno održavanje menjača obavlja na oko 45.000 km.

Redovno održavanje i servis podrazumeva zamenu ulja automatskog menjača, zamenu filtera ulja i zaptivača kartera. U poslednje vreme se sve više pojavljuju automatski menjači bez potrebe za održavanjem u predviđenom periodu eksploatacije. Praksa je u ovim slučajevima pokazala da menjači ove vrste svakako bez održavanja dosegnu svoj predviđeni životni vek, ali s druge strane redovno održavanje ipak doprinosi njegovom produženju i lepšem radu automatskog menjača.

Među modelima menjača, iako svi rade na istom principu, postoje značajne razlike, pa je za pravilno održavanje menjača uvek je potrebno pratiti uputsva data od strane proizvođača. Takođe je bitno uočiti da je moguće da isti model automobila ima drugi model menjača, stoga je pre bilo kakve intervencije potrebno prvo utvrditi tačnu marku i model menjača. U odnosu na tip menjača i uputstva proizvođača određuje se tip i količina ulja, potreban filter ulja i zaptivač kartera. U daljem tekstu poglavlja biće opisan servis menjača ZF 5HP19, na automobilu Audi A4 1.9 TDI (B5).

Prva stvar koju je potrebno uraditi je ispustiti ulje iz menjača. Da bi se olakšalo isticanje ulja,poželjno je da se ono ispušta na "vrućem motoru", što doprinosi nižoj viskoznosti i time lakšem isticanju. Za ispuštanje ulja iz menjača postoji predviđeni otvor zatvoren nekom vrstom zavrtnja. Neretko se dešava da postoje dva otvora, jedan za ispuštanje, drugi za dolivanje ulja, mada je češći slučaj da za obe svrhe postoji samo jedan otvor. Problem ispuštanja ulja kod automatskih menjača je što je u realnosti nemoguće ispustiti celokupnu količinu ulja iz menjača. Litra dve ulja, u zavisnosti od veličine, ostaje u HDTOM. Jedan deo ulja takođe ostaje zarobljen u sistemu za hlađenje, mada je ovu količinu ulja moguće ispustiti preko spoja menjača sa hladnjakom. Naravno određeni deo ulja ostaje u hidrauličkim cevovodima i elementima menjača. Ovo se svakako primeti iz činjenice da je ukupni uljni kapactet menjača ZF 5 HP 19 približno 9 litara, a prilikom servisa zamenjeno je između 6 i 7 litara.

Nakon ispuštanja ulja moguće je skinuti karter i očistiti ga od prljavština i starog zaptivajućeg silikona. Po dnu kartera raspoređeni su magneti sa skupljanje sitnih metalnih čestica iz ulja, u najčešćem slučaju nastalih trošenjem elemenata kao što su lamele spojnica i kočnica. Magnete je potrebno očistiti od metalne prašine kako bi se sprečila kontaminacija novog ulja prljavštinom.

Nakon što se na menjač se montira novi filter ulja, potrebno je karter zavrtanjskom vezom pričvrstiti za kućište. Na naleganju između kartera i kućišta postavlja se novi zaptivač i premazuje malom količinom zaptivajućeg silikona.

Sipanje ulja u automatske menjače je specifično iz nekoliko razloga. Prva stvar je staro ulje koje je ostalo u određenoj dozi zarobljeno u menjaču, čija količina varira u odnosu na više faktora, što uvodi problem potrebne količine novog ulja. Druga stvar je da se kod većine automatskih menjača otvor za dosipanje ulja nalazi sa donje strane menjača na karteru. Da bi se sipalo ulje u automatski menjač potrebna je određena vrsta "pumpe" uglavnom u vidu boce sa pumpom za vazduh. Problem sa promenjivom potrebnom količinom ulja koju treba uliti u menjač rešava se konstrukcijom kartera i otvora za dosipanje ulja. Otvor za dosipanje je napravljen iznad najniže tačke na karteru, tako da je njegovom visiom određena količina ulja koja može da bude u karteru, ulje se sipa dok ne krene da curi nazad kroz otvor. Ulje se sipa u menjač do ove tačke, nakon čega se vozilo startuje, čime se pokreće menjač i svi hidraulički sistemi u njemu. Ulje kreće da cirkuliše unutar menjača i nivo ulja u karteru se spušta. U ovom momentu se nastavlja dosipanje ulja u menjač sve do tačke kada nivo ulja u karteru opet pređe visinu otvora za dosipanje i ne krene da curi nazad. Ovim pristupom sistemi u menjaču sigurno će imati dovoljnu količinu ulja za funkcionisanje, a potrebna rezerva ulja nalazi se u karteru.

Nakon završenog servisa potrebno je putem dijagnostike resetovati ECU menjača i OBD. Vozilo se nakon toga startuje i obavlja se test vožnja, u svim modovima menjača zasebno, radi provere mogućih problema i grešaka koje mogu da se jave.

4. REPARACIJA I MOGUĆI PROBLEMI U EKSPLOATACIJI AUTOMATSKIH MENJAČA

Reparacija automatskih menjača podrazumeva uklanjanje mogućuh kvarova sistema automatskog menjača koji sprečavaju njegov pravilan rad. Uz redovano propisano održavanje preko 90% menjača nadživi svoj definisani životni vek bez potrebe za reparacijom [2], ali s drge strane zamenom "potrošenih" elemenata može se dalje produžiti životni vek menjača.

U većini slučajeva kvarova menjač počne sa malim smetnjama u toku rada, koje eksponencijalno rastu ako se problem ne ukloni. Ovakve smetnje u radu menjača se mogu uočiti tokom vožnje i ako se pojave potrebno je što pre ukloniti problem kako ne bi došlo do havarije. Najčešći problemi u radu automatskog menjača, sa njihovim uzrocima i rešenjima dati su u tabeli 1, a neki od čestih znakova probema u radu automatskog menjača su:

- nemogućnost promene SP ili promene moda menjača (D, N, R…),
- promena SP se dešava kasno ili dolazi do preskakanja nekih SP,

- proklizavanje menjača i rad motora na visokim obrtajima,
- curenje ulja iz menjača,
- miris paljenja iz menjača,
- zvuci kliktanja, zujenja i sl,
- vozilo nema pun intenzitet snage,
- paljenje check-engine lampice na OBD.

Tabela 1. Prikaz mogućih problema, uzroka i rešenja kod automatskih menjača

PROBLEM	UZROK	REŠENJE
CURENJE ULJA	Oštećen zaptivač kartera	Zamena zaptivača
	Oštećen karter	Zamena kartera
	Oštećeni semerinzi i zaptivne gumice	Zamena semeringa i zaptivnih gumica
	Curenje cevovoda ulja	Zamena cevovoda
	Otpušteni zavrtnjevi na karteru	Stezanje zavrtnjeva na karteru
	Greška na HDTOM	Zamena ili reparacija HDTOM
PREGREVANJE MENJAČA	Nedovoljno ulja u menjaču	Dosipanje ulja
	Pregorelo ili staro ulje	Zamenja ulja
	Zapušeni cevovodi ulja	Ispiranje ili zamena cevovoda
	Šlepanje, vruća klima	Obustavljanje vožnje, dodavanje fluida za hlađenje u sistem
	Defektivan solenoid	Zamena solenoida
KLIKTANJE ILI ZUJANJE	Nedovoljno ulja u menjaču	Dosipanje ulja
	Greška na HDTOM	Zamena ili reparacija HDTOM
PROKLIZAVANJE / NEMOGUĆNOST PREBACIVANJA MODA (D, N, R)	Nedovoljno ulja u menjaču	Dosipanje ulja
	Defektivan solenoid	Zamena solenoida
	Istrošeni zupčanici	Zamena zupčanika
	Potročene ferode kočnica	Zamena kočnica
	Spaljene ili potrošene lamele spojnica	Zamena lamela
ODLOŽNA ILI NEREDOVNA PROMENA SP	Nedovoljno ulja u menjaču	Dosipanje ulja
	Pogrešna vrsta ulja u menjaču	Zamena ulja sa ispravnom vrstm ulja
	Pregorelo ili staro ulje	Zamenja ulja
	Defektivan solenoid	Zamena solenoida

Kao što se može primetiti u tabeli, moguć uzrok skoro svakog kvara na automatskom menjaču je manjak ulja u menjaču ili izgorelo i prljavo ulje, ovo je uvek prva stvar koja se mora proveriti. Ulje je apsolutno neophodno za pravilan rad menjača, što sa strane podmazivanja elemenata, tako i sa strane hidrauličke kontrole i prenosa energije. Jedini način da se menjač instant uništi je rad bez ulja.

Jedan od najčešćih uzroka lošeg rada menjača su potrošene lamele spojnica i / ili kočnica, primeri prikazani na slici 1. S obzirom da i spojnice i kočnice rade na osnovu frikcije, neizbežno je da tokom rada dolazi do njihovog habanja i samim tim stanjivanja, naravno prljavština u ulju doprinosi ovom procesu. Pregrevanjem lamela dolazi i do njihovog krivljenja, dobijaju "šeširast" oblik.

Hidroblok automatskog menjača je sklop različitih ventila i delova cevovoda između njih. Radom hidrobloka ležišta njegovih pokretnih elemenata se habaju, ovo je daleko povećano u slučaju zaprljanog ulja. Ovo habanje dovodi do povećanja zazora između klipova i tela ventila, što dovodi do pada u pritisku i nepravilnog rada menjača. Ovo se u vožnji ogleda preko loše promene SP automatskog menjača. U slučaju dalje eksploatacije, zazori se povećavaju sve do havarije menjača. Reparacija hidrobloka je moguća, ali je daleko bolje i relativno jeftinije rešenje kupovina novog zamenskog hidrobloka.

Slika 1. *Primer ekstremno potrošenih lamela spojnice*

Proklizavanje spojnica i loša ili nemoguća promena SP takođe mogu biti prouzrokovani lošim radom ventila sa solenoidima. U slučaju kvara solenoida ventil ostaje u nekom od svojih položaja i njegova kontrola na dalje nije moguća, što dalje znači da menjač neće moći da radi. U

slučaju lošeg rada solenoida, ne postoji opcija remonta, nego se uvek kupuju i ugrađuju novi zamenski solenoidi. Kako svaki menjač poseduje više ovakvih ventila, preporuka proizvođača je da se uvek menjaju svi solenoidi u isto vreme iako je greška samo u jednom.

Pogrešan rad HDTOM može se prepoznati kliktanjem ili zujanjem menjača u toku svog rada. Naravno većina grešaka u radu mogu se detektovati i putem eksterne dijagnostike. Radom menjača dolazi do habanja elemenata HDTOM kao i propadanja spojnica u njemu. Prosečan životni vek HDTOM je oko pređenih 200.000 km. Veliki problem, kada je ovaj element u pitanju, je činjenica da je sklop HDTOM nerastavljiv.

Da bi se uradio remont HDTOM potrebno je preseći njegovo kućište, zameniti lamele spojnice, i sastaviti ga opet putem zavarene veze. Prilikom sklapanja HDTOM potrebno je posebnu pažnju obratiti na saosnost dve polovine, u suprotnom dolazi do bacanja u toku rada HDTOM i na kraju do havarije. Drugi način rešavanja problema je naravno zamena starog sa novim fabričkim HDTOM.

Kad su u pitanju elementi mehaničkog sistema kao što su vratila, zupčanici, ležajevi i slično, retko kada dolazi do problema. Moguće je da mehanički elementi prenosa ne obavljaju svoj posao korektno, ali uglavnom je to samo posledica lošeg rada nekog od ostalih sistema.

Do lomova zubaca zupčanika ili drugih mehaničkih elemenata dolazi u slučaju fabričke greške, uglavnom zbog grešaka u materijalu elementa, prevelikog zazora u ležajevima ili u slučaju jako neodgovornog korišćenja automatskog menjača.

5. ZAKLJUČAK

Kao i kod svih mašina i mehanizama, redovno održavanje je ključno za pravilno funkcionisanje sistema. Shodno uskim tolerancijama sklopova menjača, kao i funkcionisanju hidrauličkog sistema, loš kvalitet ulja, njegova zaprljanost ili manjak ulja ekstremo loše utiču na rad i životni vek menjača. Loš kvalitet ulja ili neodgovarajuća vrsta ulja intenziviraju trošenje elemenata menjača, u nekim slučajevima čak dolazi i do apsolutnog prekida rada menjača. Zaprljanost ulja sitnim metalnim česticama dovodi do habanja hidrauličkih elemenata, povećanju njihovih zazora i time do kolapsa sistema.

Prljavo ulje ubrzava trošenje frikcionih elemenata kočnica i spojnica i dovodi do prerane podrebe za njihovom zamenom. Nedovoljna količina ulja dovodi do lošeg podmazivanja, umanjenog pritiska u hidrauličkom sistemu i time do lošeg rada menjača. Mala količina ulja ili staro i zaprljano ulje eksponencijalno ubrzavaju propadanje menjača i u ekstremnim situacijama dovode do havarije menjača.

Automatski menjač je umrežen sa ostalim sistemima i podsistemima vozila i u slučaju njegovog lošeg rada uvek se treba prvo uveriti i u ispravnost ostalih sistema pre analize samog menjača. Kako je menjač mehatronički sistem, detekcija uzroka problema se u najvećoj meri svodi na sisteme eksterne dijagnostike, bez koje nije moguće izvršiti reparaciju. S druge strane uz ovo, potrebno je veliko iskustvo kako bi se uz dijagnostiku pravilno detektovao i otklonio uzrok problema.

Reparacijom automatskih menjača uveliko im se produžava životni vek. Automatski menjači na putničkim vozilima, uz reparaciju, mogu da pređu i preko 400.000 km što je više nego duplo od njihovog projektovanog životnog veka. Zamesnski delovi automatskih menjača (lamele spojnica / kočnica, solenoid itd) lako mogu da se nađu na tržištu, po kataloškom broju proizvođača. S druge strane zamena ovakvih delova je vrlo odgovoran posao, pa cene reparacije menjača u nekim slučajevima starijih vozila znaju da prevaziđu cenu vozila i postaju neisplative.

Praksa je pokazala da su havarije kod automatskih menjača vrlo redak slučaj. Do havarija dolazi zbog trenutnog gubitka ulja u menjaču u toku njegovog rada ili ekstremno neodgovorne vožnje neodržavanog menjača. Do trenutnog gubitka ulja u menjaču može doći samo prilikom udara i fizičkog loma kućišta ili kartera menjača. U drugim slučajevima do havarija generalno dolazi nakon što je menjač prevazišao svoj životni vek, a nije na vreme odrađena potrebna reparacija menjača. Može se zaključiti da su automatski menjači, iako kompleksne konstrukcije, u eksploataciji veoma pouzdani i robusni, sa pozitivnim uticajem na ukupne performanse vozila, udobnos i sigurnost u vožnji.

6. LITERATURA

[1] Yong Chen, *"Automotive Transmissions – Design, Theory and Applications"*, Springer, 2021.

[2] H. Naunheimer, B. Bertsche, J. Ryborz, N. Wolfgang, *"Automotive Transmissions – Fundamentals, Selection, Design and Application"*, Springer, 2011.

[3] R. Ficher, F. Kucukay, G. Jurgens, R. Najork, B. Pollak, *"The Automotive Transmission Book"*, Springer, 2015.

[4] B. Stojić, N. Poznanović, D. Ružić, J. Dorić, "*Drumska voila*", Novi Sad, FTN

Kratka biografija:

Srđan Nikačević rođen je u Novom Sadu 1995. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Mašinskog inžinjerstva – Razvoj protetičke noge koja omogućuje osobama sa amputacijom vožnju motocikla odbranio je 2019.god. kontakt: se.nika@uns.ac.rs

dr Jovan Dorić rođen je u Novom Sadu 1983. god. Doktorirao je na Fakultetu tehničkih nauka 2012. god, a od 2017 je zvanju vanredni profesor. Oblast interesovanja su motorna vozila i motori SUS.
kontakt: jovan_d@uns.ac.rs

dr Marko Vilotić rođen u Novom Sadu 1979. god. Diplomu osnovnih i magistarskih studija, kao i doktorat stekao je na Fakultetu tehničkih nauka. Na Fakultetu tehničkih nauka (Katedra za tehnologije oblikovanjem i inženjerstvo površina) radi kao docent.
kontakt: markovil@uns.ac.rs

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.7/.9
DOI: https://doi.org/10.24867/15AM02Bozic

INFORMACIONI SISTEM ZA PRAĆENJE TOKA ALATA U MALOM PROIZVODNOM PREDUZEĆU

INFORMATION SYSTEM FOR MONITORING TOOL FLOW IN THE SMALL PRODUCTION ENTERPRISE

Dejan Božić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj - *Informacioni sistemi u proizvodnim pogonima malih i srednjih preduzeća značajno mogu da utiču na povećanje produktivnost i ekonomičnosti. Osnovni predmet i cilj istraživanja u ovom radu se odnosi na istraživanje vremenskih gubitaka usled nepostojanja informacionog sistema vezanog za praćenje toka alata, kao i prikaz idejnog rešenja ovog sistema.*

Ključne reči: *Informacioni sistem, Vremenski gubici, Tok alata, Malo proizvodno preduzeće*

Abstract - *Information systems in the production facilities of small and medium enterprises can significantly increase productivity and economy. The main subject and goal of the research in this paper is to investigate time losses in the midst of the lack of information system related to monitoring the flow of tools, as well as the presentation of the conceptual solution of this system.*

Keywords: *Information system, Time loses, Tool flow, Small production enterprises*

1. UVOD

"JOMI-ING" je malo preduzeće koje prati savremeni tehnološki napredak, kroz nabavku modernih mašina renomiranih svetskih proizvođača, kao i visokokvalitetnih alata, pribora i mernih sistema, čime se izvršavaju brojni izuzetno složeni i precizni zadaci u skladu sa zahtevima kupaca. Proizvodni program čini širok spektar izrade delova i montaže sklopova za razne industrijske potrebe: autoindustriju, farmaceutsku i optičku industriju [1].

Kao predmet istraživanja nameće se sistematizacija i unapređenje proizvodnje posmatranog preduzeća, gde se na prvo mesto postavlja razvoj i implementacija informacionog sistema koji je prilagođen potrebama ovog malog preduzeća. Osnovni cilj ovog rada odnosi se na postavku idejnog rešenja informacionog sistema za praćenje toka alata kao jedne od vitalnih aktivnosti za unapređenje poslovanja posmatranog preduzeća.

2. INFORMACIONI SISTEMI U PREDUZEĆU

2.1. Osnove informacionog sistema

Kvalitetno rukovođenje preduzećem nije moguće zamisliti bez neophodnog resursa koji predstavlja informacija. Dobra i pravovremena odluka iziskuje da informacija mora biti tačna, precizna i u pravo vreme.

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

Informacioni sistem predstavlja sistem koji će na osnovu formalizovanih procedura obezbediti upravljanje na bazi informacija iz internih i eksternih izvora, a te odluke će se koristiti za potrebe planiranja, rukovođenja i kontrole aktivnosti. Informacioni sistem u širem smislu predstavlja kompletan sistem protoka, razmene i obrade informacija. Takođe, ovaj pojam se smatra širim od pojma informacione tehnologije, a zanimljivo je da čak i ne mora biti baziran na računarskoj tehnologiji. Informacioni sistem predstavlja skup organizovanih komponenti koje omogućuju [2]: Registrovanje; Prikupljanje; Prenos; Obradu; Skladištenje; Analizu i Distribuciju informacija za različite namene.

Informacioni sistem, takođe, može da se predstavi kao organizovani skup međusobno povezanih komponenti, kao što su: Softverski resursi; Hardverski resursi; Mrežni resursi; Resursi podataka i Ljudski resursi.

Pod softverskim resursima podrazumevaju se programi i procedure, hardverski resursi obuhvataju računarske uređaje i medije, mrežni resursi komunikacione medije i računarske mreže, resursi podataka obuhvataju baze podataka i znanja, a ljudski resursi uključuju specijaliste i korisnike informacionih tehnologija.

2.2. Motivi implementacije informacionih sistema

Preduzeća se moraju konstantno razvijati kroz unapređenje znanja, praćenje trendova i noviteta. Ključ uspeha predstavlja želja za uspehom, razumevanje noviteta i implementacija isključivo optimalnog odnosa noviteta u postojeće preduzeće koje će jasno unaprediti poslovanje i olakšati zaposlenima određene aktivnosti. Neki od motiva koji odlučuju da se izvrši implementacija informacionog sistema mogu biti sledeći:

- *Integrisanje informacija iz odnosa sa kupcima, posebno u vezi sa porudžbinama kupaca*
- *Standardizovanje i ubrzavanje proizvodnih procesa*
- *Integrisanje finansijskih informacija*
- *Redukovanje zaliha*

2.3. Načini implementacije informacionih sistema

Implementacija ovakvih rešenja može da traje od jedne do pet godina, iako je taj termin značajno kraći ukoliko dobavljači predstavljaju vremenski zahtev implementacije. Postoje sledeći načini implementacije:

BigBang strategija – Ova strategija podrazumeva da se u istom momentu pređe sa jednog na drugi način poslovanja u skladu sa ERP zahtevima. Zahteva se da kompletan ERP sistem bude do detalja urađen i da se po prelasku na taj sistem koristi u punom kapacitetu. Upravo zbog toga

se dešavaju propusti koji se moraju ispraviti u što kraćem roku. Uspeh implementacije ERP sistema na ovaj način je vidljiv tek nakon nekoliko godina.

Franchising strategija - Karakteristična je za velike korporacije gde postoji mreža sa udaljenim poslovnim jedinicama koje poseduju visok stepen autonomije upravljanja. Franchising strategija podrazumeva instalaciju, podešavanje i puštanje u rad jedne licence ERP sistema u određenoj poslovnoj jedinici. Ako implementacija prođe uspešno, ona postaje referentna za nove implementacije u ostalim poslovnim jedinicama.

Slam Dunk strategija – Ova strategija ima za cilj implementaciju informacionog sistema samo u okviru ključnih procesa poslovnog sistema. Na osnovu toga pri implementaciji ERP sistema, takav sistem se bira i prilagođava već postojećim poslovnim procesima. Pogodnosti strategije su što je niskog rizika upravo zbog toga što se realizuje postepeno, pa se stoga ne vrši značajan reinženjering, već se samo uvodi novi alat za podršku poslovnim procesima.

3. TOK INFORMACIJA I ALATA U PREDUZEĆU

3.1. Strategije proizvodnje u preduzeću

Osnovne strategije proizvodnje prema uticaju kupca na proizvod, možemo podeliti na sledeći način [3, 4]:

Proizvodnja za zalihe – MTS (Make to stock) – Predstavlja strategiju kod koje se gotovi proizvodi skladište i čekaju kupca. Karakteristike strategije su kratko vreme od narudžbine do isporuke, ali su vreme i troškovi skladištenja veliki, dok je uticaj kupca na karakteristike proizvoda mali.

Montaža prema narudžbi – ATO (Assembly to order) – Predstavlja strategiju kod koje proizvođač vrši montažu gotovih proizvoda koje čine modularni delovi i podsklopovi prema zahtevu kupca.

Proizvodnja prema narudžbi – MTO (Make to order) – Predstavlja strategiju kod koje proizvođači čekaju narudžbu od kupca da bi prilagodili proizvod koji se sastoji od gotovih modularnih komponenti i komponenti koje je tek potrebno projektovati i izraditi.

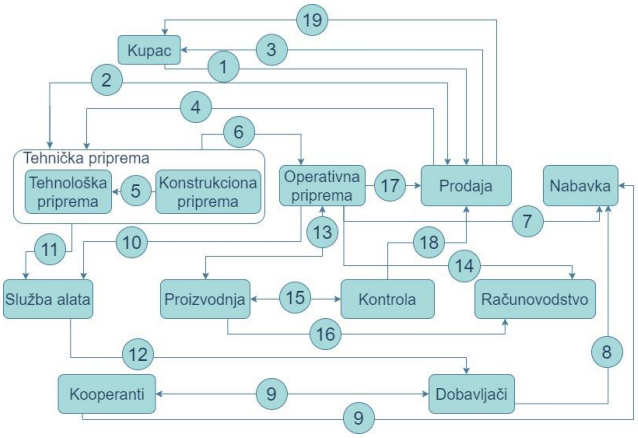
Inženjering prema narudžbi – ETO (Engineering to order) – Predstavlja strategiju kod koje kupac definiše specifikaciju i funkcionalnost proizvoda, a proizvođač projektuje i proizvodi odgovarajući proizvod prema dostavljenoj specifikaciji.

Preduzeće je najviše orijentisano na proizvodnju/pružanje usluga izrade delova i sklopova prema narudžbi i tehničkoj dokumentaciji kupca. Shodno tome u nastavku će se razmatrati organizacija rada službi preduzeća i tokovi informacija prema ovoj strategiji poslovanja.

3.2. Organizaciona struktura preduzeća

Cilj kojem teže preduzeća koja su pretežno orijentisana ka pojedinačnoj i maloserijskoj proizvodnji, predstavlja ispunjenje zahteva kupca. Zbog jasnijeg prikaza funkcionisanja posmatranog preduzeća na slici 1. dat je tok praćenja narudžbi u preduzeću, od momenta dobijanja zahteva od kupca pa sve do isporuke proizvoda [5].

U okviru tabele 1 detaljno su objašnjeni koraci toka informacija sa slike 1, gde su predstavljene i informacije o toku alata u preduzeću. Informacije o toku alata vezane su za više odeljenja preduzeća, tako da je to ujedno i uvod za dalju analizu toka podataka o alatima.



Slika 1. *Tok aktivnosti u preduzeću JOMI-ING*

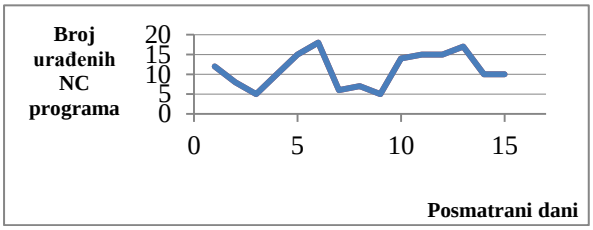
Tabela 1. *Aktivnosti pri praćenju narudžbina*

TOK	AKTIVNOST
1	Kupac kontaktira preduzeće (prodaju) kako bi poručio proizvod prema svojim zahtevima.
2	Predaja dostavlja zahtev/narudžbu kupca službi tehničke pripreme proizvodnje. Konstrukciona priprema proizvodnje razmatra novitet proizvoda i da li je potrebna njena aktivnost. Ukoliko narudžbina sadrži crteže proizvoda prosleđuje se u tehnološku pripremu proizvodnje. Nakon toga, vrše se konsultacije između tehnološke pripreme proizvodnje, operativne pripreme proizvodnje i nabavke u cilju definisanja mogućnosti izrade, kapaciteta, grubih rokova isporuke i procene troškova – cene isporuke.
3	Prodaja šalje kupcu ponudu (koncept proizvoda – ukoliko postoji takav zahtev, rokove isporuke i cenu) na odobravanje. Ukoliko se kupac slaže sa ponudom, potpisuje se ugovor o izradi i isporuci proizvoda.
4	Kada se ugovor potpiše, prodaja izdaje tehničkoj pripremi nalog. Tehnička priprema, operativna priprema proizvodnje i nabavka usaglašavaju rokove (nabavke, završetka konstrukcione i tehnološke dokumentacije, početka i završetka proizvodnje i termin isporuke proizvoda).
5	U okviru tehničke pripreme proizvodnje, konstrukciona priprema proizvodnje šalje tehnološkoj pripremi proizvodnje konstrukcionu dokumentaciju. Vrši se izrada i dopuna tehnološke dokumentacije (sadržaj tehnološkog procesa, karte operacija, upravljački programi…). Definisanje troškova proizvodnje.
6	Predaja konstrukcione i tehnološke dokumentacije u službu operativne pripreme proizvodnje.
7	Provera stanja zaliha pripremaka, komponenti (delova i sklopova) i dr. Slanje zahteva za nabavku pripremaka i komponenti službi nabavke.
8	Kontakt sa dobavljačima i odgovor o rokovima i količinama isporuke pripremaka, komponenti (delova i sklopova) i dr.
9	Kontakt sa kooperantima i odgovor o rokovima i količinama dostave robe.
10	Provera ispravnosti postojećih alata, pribora, merila, uređaja i dr. i narudžba za popravku istih, izradu ili poručivanje novih.
11	Tehnička dokumentacija (konstrukciona i tehnološka dokumentacija) specijalnih alata, pribora, merila, uređaja i dr.
12	Porudžba alata, pribora, merila i uređaja od dobavljača
13	Lansiranje konstrukcione dokumentacije, tehnološke dokumentacije i radnih naloga u proizvodnju. Povratne informacije o realizaciji radnih naloga iz proizvodnje u operativnu pripremu proizvodnje.
14	Lansiranje dokumentacije iz operativne pripreme proizvodnje u računovodstvo.
15	Komunikacija između proizvodnje i kontrole o ispravnosti izrađenih proizvoda.
16	Lansiranje proizvodne dokumentacije u računovodstvo.
17	Kontinualne informacije o prodaji iz proizvodnje o stanju dovršenosti proizvoda, poteškoćama, eventualnim kontaktima sa kupcima i dr.
18	Obaveštenje službe prodaje o završetku proizvodnje.
19	Isporuka gotovih proizvoda kupcu.

3.3. Vremenski gubici programera

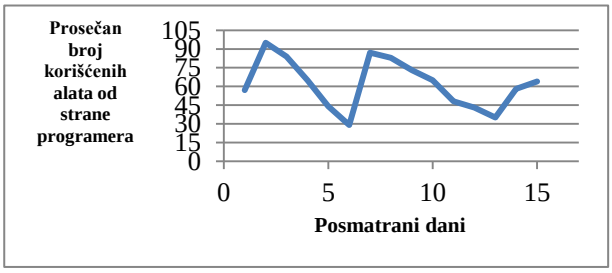
Razvoj informacionog sistema za praćenje alata uslovljen je vremenskim gubicima programera, te je zbog toga dat uvid u kriterijume njihovog nastanka. Posmatrani su razni kriterijumi u toku od tri radne nedelje (15 radnih dana), koji su dati u nastavku.

Kao prvi parametar za uvid u vremenske gubitke programera izdvaja se broj kreiranih NC programa na dnevnom nivou, slika 2.



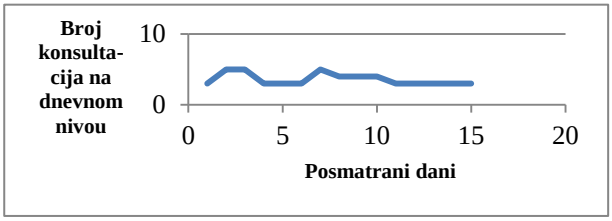
Slika 2. *Broj urađenih NC programa na dnevnom nivou za vremenski period od tri nedelje*

Kao sledeći parametar uzet u razmatranje jeste broj alata koji se koriste na dnevnom nivou. Na slici 3 dat je uvid u dnevni prosek korišćenja alata (brojčane vrednosti).



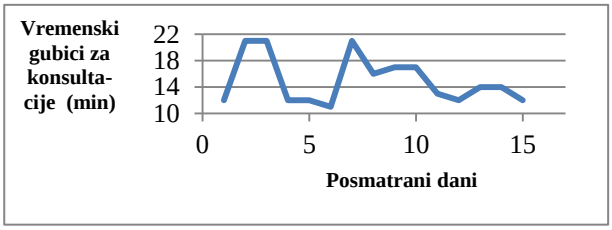
Slika 3. *Prosečan broj dnevno korišćenih alata za vremenski period od tri nedelje*

Slika 4 pokazuje koliko je programer puta imao konsultacije vezano za određeni alat, ili je sam tražio alat u alatnici, na dnevnom nivou.



Slika 4. *Broj konsultacija vezane za alate na dnevnom nivou za vremenski period od tri nedelje*

Na slici 5. prikazani su vremenski gubici programera. Vremenski gubici variraju, a prosek posmatrane tri nedelje je oko 15 minuta dnevno. U posmatranom periodu od tri nedelje izgubljeno je ukupno 225 minuta sa stanovišta programera, što jeste ozbiljan gubitak, pogotovo ako se uzme u obzir da u pogonu ima više programera mašina.



Slika 5. *Prikaz vremenskih gubitaka programera*

4. IDEJNO REŠENJE INFORMACIONOG SISTEMA ZA PRAĆENJE TOKA ALATA

Pošto se informacije vezane za tok alata, u posmatranom preduzeću tiču menadžera kvaliteta i nabavke, programera i šefa proizvodnje (gde se podrazumevaju i operateri), prikazani koncept rešenja biće predstavljen iz njihovog ugla. Na samom startu definisane su informacije baza podataka za alate, mašine i operatere. Aplikacija koja je korišćenja za formiranje elemenata baze podataka je MS Excel, a aplikacija u okviru koje su kreirani različiti interfejsi kao i povezivanje baza podataka je MS Access.

Na slici 6. prikazane su inormacije baze podataka alata, u okviru koje su trenutno uneti opšti podaci.

Ovu bazu podataka informacijama pohranjuje menadžer za kvalitet i nabavku. Ako se kupi novi alat, on ima dužnost da informaciju unese u bazu, tako što upisuje sve potrebne podatke ili u koloni "Količina" ažurira novo brojno stanje u skladu sa brojem novih alata i trenutno postojećih u okviru preduzeća. Pod kolonom "Tip alata" podrazumeva se vrsta alata, pa je zbog lakše pretrage prvo je stavljena vrsta alata, pa tek onda podvrsta (kao na primer glodalo_vretenasto ili glodalo_ceono).

U koloni "Prečnik" upisuje se prečnik alata, dok se u kolonu "Dubina" unosi maksimalna dubina koju alat može da obrađuje. Kolone "M_A" i "M_O" predstavljaju materijal alata i materijal obratka, koji su značajni ako se koristi iMachining u okviru SolidCam - a gde se unošenjem ovih podataka, softverskim putem određuju optimalni režimi obrade prema kriterijumu maksimalnog kvaliteta obrađene površine ili maksimalne količine skinute strugotine. Kolone "Brz. Rezanja", "Pomak" i "Dubina" predstavljaju preporučene režime obrade prema katalozima proizvođača alata.

	B	C	D	E	F	G	H	I	J
	Tip alata	Precnik	Dubina	M_O	Kolicina	M_A	Brz. Rezanja	Pomak	Dubina
1	Burgija	7	8	AL	7	TM	100	0,1	1
2	Burgija	10	14	AL	11	TM	100	0,1	1
3	Burgija	8	10	AL	5	TM	100	0,1	1
4	Burgija	12	15	AL	7	TM	100	0,1	1
5	Glodalo_vretenasto	8	9	AL	5	TiAlN	100	0,1	1
6	Glodalo_vretenasto	12	12	AL	9	TiAlN	100	0,1	1
7	Glodalo_vretenasto	20	20	AL	7	TiAlN	100	0,1	1
8	Glodalo_vretenasto	15	15	AL	8	TiAlN	100	0,1	1
9	Glodalo_vretenasto	10	10	AL	11	TiAlN	100	0,1	1

Slika 6. *Prikaz elemenata baze podataka alata*

Baza podataka operatera sadrži ID operatera i bitna je sa aspekta praćenja mesta lokacije preuzetog alata.

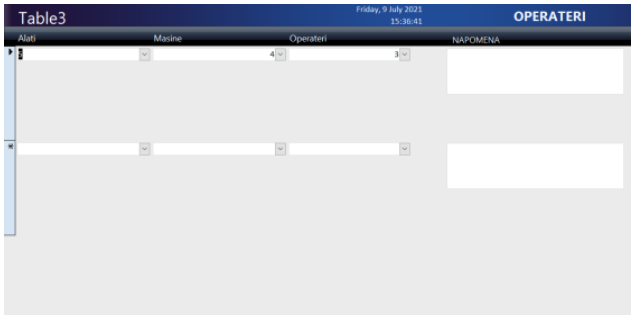
Baza podataka mašina, sadrži 8 obradnih centara za glodanje (Obradni centar G) i 3 obradna centra za struganje (Obradni centar S). Pored naziva mašine u bazi se nalaze geometrijske karakteristike mašina ("X osa","Y osa" i "Z osa") i maksimalni broj obrtaja glavnog vretena mašine.

Razliku interfejsa prave različite potrebe zaposlenih u tom smislu, da ako na primer programer želi da izabere mašinu na kojoj će se vršiti obrada, on treba da zna i karakteristike mašine kako ne bi kreirao i poslao NC program na obradni centar koji zbog dimenzija radnog prostora nije odgovarajući. Sa druge strane, ako operater uzima alat iz alatnice po nalogu programera - tehnologa, njemu je važno da ubeleži na kom obradnom centru će se nalaziti alat.

Interfejs operatera

Na slici 7. prikazan je interfejs operatera za preuzimanje alata iz magacina. Dužnosti operatera su da izabere iz padajućeg menija polja koja se odnose na alat koji bira, zatim na koju mašinu nosi alat i na kraju koji operater je

zadužio alat. Ukoliko postoji potreba za nekom napomenom unosi ga u polje sa desne strane. Ovaj interfejs automatski beleži vreme kada je operater preuzeo alat.
Isti interfejs se koristi i kod vraćanja alata, s tim što postoji dodatan deo "Lom alata", gde je potrebno uneti reč "DA" ukoliko je došlo do loma alata (da bi se pravovremeno poručila zamena) ili uneti reč "NE" ukoliko nije došlo do oštećenja. U okviru "NAPOMENA" poželjno je rečima opisati ukoliko je došlo do sitnijih oštećenja alata ili slično.

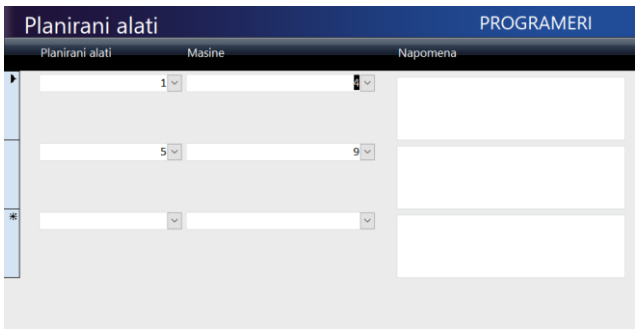


Slika 7. *Izgled interfejsa za preuzimanju alata*

Interfejs menadžera kvaliteta i nabavke
Interfejs za uvid u raspoloživost alata po pitanju stavki "Alati" i "Mašine" su identični sa prethodno opisanim slikama. Dodatna polja predstavljaju "Količina" i "Raspoloživo", slika 8. Na osnovu izbora alata, automatski se popunjavaju sledeća polja, pa se tako dobija "Količina" pod kojom se podrazumevaju svi alati koji se nalaze u preduzeću (koji se trenutno koriste i koji se ne koriste). Ovaj broj je važan kako bi menadžer kvaliteta i nabavke, u slučaju manjeg broja alata, pravovremeno naručio dodatne količine. Pod stavkom "Raspoloživo" podrazumeva se broj alata koji se trenutno ne koriste na mašinama, a raspoloživi su za korišćenje.

Slika 8. *Interfejs za uvid u raspoloživost alata*

Interfejs programera
Na slici 9. prikazan je izgled interfejsa planiranih alata koji koristi programer. U okviru njega može se birati tačno određeni alat i mašina na kojoj se planira koristiti alat, kao i dodatna napomena ukoliko postoji potreba. Padajući meniji pod stavkama "Planirani alati" i "Mašine" su identični sa opisom prethodnih. Ovaj deo je važan sa aspekta da označavanjem planiranih alata za upotrebu, drugi programer vidi podatak i u skladu sa tim planira alate.



Slika 9. *Interfejs planiranih alata*

5. ZAKLJUČAK

Svako preduzeće teži modernizaciji, pre svega kroz primenu savremenih informaciono-tehnoloških unapređenja, uz minimalni stepen nesigurnosti razvoja, što na kraju daje višegodišnje uspešno poslovanje. Vreme pokazuje da se dešavaju radikalne promene u načinu poslovanja, a te promene dolaze sve brže. Njihovo neprihvatanje ili potpuno odbacivanje smanjuje konkurentnost preduzeća na tržištu, jer poslovanje u novije vreme postaje sve transparentije, pa se lakše može zaključiti na koji način preduzeće funkcioniše.
Cilj ovog rada je ispunjen tako što je detaljno prikazan problem upravljanja alatima u preduzeću, a rešenje je prikazano u obliku ideje, odnosno koncepta na koji bi se način taj problem mogao rešiti. Unapređenje u komunikaciji zaposlenih pri upravljanju alatima u preduzeću zauzvrat daje velike vremenske i novčane uštede. Zbog toga se nameće logičan zaključak da je ovakav sistem realno potrebno razviti i implementirati u posmatrano preduzeće.

6. LITERATURA

[1] http://www.jomi-ing.com/saboutUs.html (pristupljeno u avgustu 2021.)
[2] D. Berić, "*Model informacionog sistema za podršku upravljanju industrijskim preduzećima"*, doktorska disertacija, Fakultet tehničkih nauka, Novi Sad, 2019.
[3] D. Lukić, M. Milošević, V. Todić, "*Integrisani CAPP sistemi i tehnološka baza podataka",* skripta, Fakultet tehničkih nauka, Novi Sad, 2013.
[4] M. Stefanović, *„CIM sistemi"*, Mašinski fakultet, Kragujevac, 2006.
[5] D. Božić, "*Informacioni sistem za praćenje toka alata u malom proizvodnom preduzeću*", master rad, Fakultet tehničkih nauka, Novi Sad, 2021.

Kratka biografija:

Dejan Božić rođen je u Bačkoj Topoli 1997. god. Master rad na Fakultetu tehničkih nauka iz oblasti Mašinstva – Proizvodno mašinstvo, smer Računarom podržane tehnologije odbranio je 2021. god. kontakt: bbozic997@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 697.1
DOI: https://doi.org/10.24867/15AM03Lazic

PREGLED PRIMENE TEHNOLOGIJA KOGENERACIJE I SKLADIŠTENJA ENERGIJE

REVIEW OF THE APPLICATION OF COGENERATION AND ENERGY STORAGE TECHNOLOGIES

Zorana Lazić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ENERGETIKA I PROCESNA TEHNIKA

Kratak sadržaj – *Rad daje opis tehnologije, kogeneracije i skladištenja energije, na primerima dobre prakse sa ciljem povećanja efikasnosti i smanjenju zagađenja životne sredine, koje je sve veće zbog velikih svetskih potreba za energijom.*

Ključne reči: *Kogeneracija topotne i električne energije (CHP), Skladištenje energije komprimovanog vazduha (CAES)*

Abstract – *The paper describes technology, cogeneration and energy storage, on the examples of good practice with the aim of increasing efficiency and reducing environmental pollution, which is increasing due to high global energy needs.*

Keywords: *Combined head and power (CHP), Compressed Air Energy Storage (CAES)*

1. UVOD

Sve veća zabrinutost zbog uticaja energetskih sistema na životnu sredinu o svesti i važnosti odgovorne upotrebe fosilnih rezervi, doprinosi mnogim naporima da se energetski sistem preusmeri na drugačiji i održiviji energetski model. S tim u vezi, preduzete su mnoge značajne akcije kojima se podstiče upotreba obnovljive energije i povećanje efikasnosti čitavog energetskog lanca od proizvodnje električne energije do potrošnje krajnjih korisnika. Jedno od rešenja jeste upravo upotreba kogeneracije u kombinaciji sa skladištenjem energije.

Kogeneracija i skladištenje veoma su zastupljene tehnologije širom sveta, ali samo kao dva odvojena postrojenja i kao takvi su opisani detaljno u radu, i pored toga su dati neki od primera. Međutim, cilj rada jeste da se pokaže rešenje, odnosno postrojenje u kome se kombinuju kogeneracija i skladištenje, koje bi mnogo više doprinelo efikasnosti i smanjenju zagađenja životne sredine. Ovaj tip postrojenja nije mnogo rasprostranjen, ali su data dva primera gde je detaljno prikazan princip rada ovog tipa postrojenja.

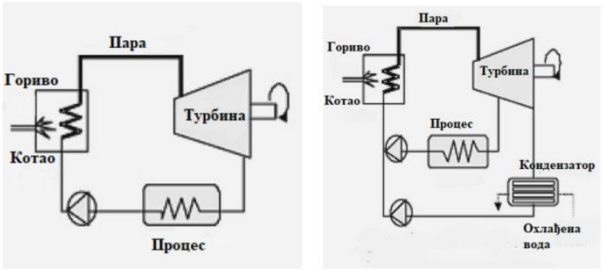
2. KOGENERACIJA

Kogeneracija je tehnologija za racionalno korišćenje energije i definiše se kao istovremeno generisanje dva ili više oblika energije iz jednog izvora. Kod kogeneracije efikasnost sistema se povećava za 80% do 95% u odnosu na konvekcionalne termoelektrane (35% do 40%).

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Aleksandar Anđelković, red.prof.

2.1. Tehnologije kogenerativnih postrojenja

Postoji mnogo vrsta kogenerativnih sistema, međutim, najčešće korišćeni sistemi su sistemi za kogeneraciju parnih turbina (STCS). Oni rade na Rankineovom ciklusu i vrlo su fleksibilni u snabdevanju gorivom. U osnovi mogu da koriste dve vrste goriva koji se mogu sagoreti u kotlu. U kogeneraciji se koriste dve vrste parnih turbina, turbina sa povratnim pritiskom i ekstrakciono-kondenzaciona turbina. Princip rada ove dve vrste turbina dat je slikom 1. gde je na levom delu slike prikazana turbina sa povratnim pritiskom, dok je na desnom delu ekstrakciono-kondezaciona turbina.



Slika 1. Princip rada turbine sa povratnim pritiskom i ekstrakciono-kondezacione turbine

Pored pomenutih tehnologija, postoji i sistem na gasne turbine (GTCS). Ti sistemi su doživeli brz razvoj poslednjih godina i postali su alternative kogenerativnim sistemima parnih turbina. Ukoliko je potrebno više električne energije, moguće je koristiti kombinovani ciklus gasa i pare (CGSC).

Regeneracijski sistem kogeneracije motora (RECS) je još jedna tehnika kogeneracije. Ovaj sistem koristi motore sa unutrašnjim sagorevanjem za obezbeđivanje obrtnog momenta i toplote sa efikasnošću većom nego u parnim i gasnim turbinama.

Pored pomenutih vrsta kogenerativnih sistema mogu se svrstati i mikroturbine i gorivne ćelije. Mikroturbine su male trubine koje sagorevaju gasovita ili tečna goriva da bi pogonila vratilo generatora, pri čijem se sagorevanju takođe oslobađa i određena količina toplote koja se koristi u sistemu kogeneracije. Ovo je relativno nova tehnologija.

Gorivne ćelije imaju znatne prednosti u odnosu na ostale tehnologije iz razloga što pružaju potencijal za čisto, nečujno i efikasno stvaranje pre svega električne, a zatim i toplotne energije. Pogonsko gorivo ne sagoreva, već reaguje elektro-hemijski, pa samim tim se postižu minimalna zagađenja vazduha. Gorivne ćelije su veoma slične baterijama po tome što i jedne i druge proizvode jednosmernu

struju kroz elektro-hemijski proces, ali bez direktnog sagorevanja pogonskog goriva.

2.2. Primena kogenerativnih postrojenja

Postoje tri vrste kogeneracije podeljene prema upotrebi. Prva je industrijska kogeneracija koja postoji u fabrikama gde je potrebna toplotna i električna energija. Druga vrsta je kogeneracija grejanja. Takvi sistemi se primenjuju kada postoji velika potražnja za toplotom, a proizvodnja električne energije je samo nusproizvod, na primer u daljinskom grejanju. Poslednji tip kogeneracije je poljoprivredna kogeneracija koja se može koristiti za ruralne procese. Korišćenjem tehnologije kogeneracije u ovoj oblasti, npr. pogoni za preradu hrane mogu koristiti nusproizvode biomase za proizvodnju topline i energije, koja se zauzvrat može koristiti za proizvodni proces.

3. SKLADIŠTENJE ENERGIJE

Trenutni zahtevi potrošača u snabdevanju električnom energijom predstavljaju osnovni uzrok složenosti elektroenergetskog sistema. Sve veća odstupanja potrošnje električne energije od njene proizvodnje dovode u pitanje stabilnosti rada celog sistema što za posledicu ima odstupanje napona i frekvencije od njihovih nominalnih vrednosti. Pojam „skladištenje energije“ se odnosi na transformaciju nekih polaznih oblika energije u obliku pogodan za skladištenje i ponovo u povratnu transformaciju. Energija se skladišti u intervalima kada proizvodnja energije nadomašuje njenu potrošnju, a skladištene rezerve se koriste kada potrošnja energije nadomapuje njenu proizvodnju. Na ovaj način se održava balans.

3.1. Vrste i karakteristike sistema za skladištenje

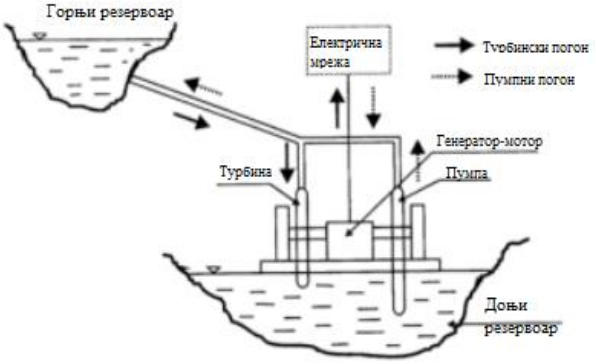
Skladištenje energije se može vršiti preko kombinovanih ili hibridnih sistema. Pod kombinovanim sistemima se smatraju sistemi koji na ulazu imaju jedan primarni izvor energije, dok na izlazu imaju dva ili više oblika energije. Dok se pod hibridnim sistemom smatraju sistemi koji imaju dva ili više ulazne energije i samo jednu izlaznu energiju. Delovi sistema za skladištenje energije su: sistem za transformaciju snage (STS), centralno skladište (CS) i upravljački sistem za punjenje i pražnjenje (USPP).

Osnovne karakteristike sistema za skladištenje su: energetski kapacitet skladištenja, maksimalna snaga i vremenska konstanta, gustina energije i gustina snage (energija i specifična snaga), trajanje ciklusa skladištenja, brzina (snaga) punjenja i pražnjenja, gubici energije i efikasnost skladištenja (efikasnost ciklusa), starenje sistema, vreme odziva, troškovi ulaganja, trajnost, ekonomičnost, kompatibilnost, autonomnost, pouzdanost, uticaj na životnu sredinu (okolinu) i sl. Ovi parametri služe za poređenje različitih metoda za skladištenje energije.

3.2. Tehnologije skladištenja energije

Koliko je proces skladištenja energije značajan u današnjem modernom svetu, pokazuje i broj tehnologija koje su se razvile kako bi se ovaj proces obavljao. Međutim, postoji četiri glavne kategorije: potencijalna mehanička energija (hidroelektrična brana, reverzibilna hidroelektrana / morska reverzibilna hidroelektrana / skladištenje energije vazduhom), kinetička mehanička energija (inercioni zamajac), elektro- hemijska energija (baterije, akumulatori, kondenzatori, vodonikovi vektori) i termalna energija (latentna ili osetna toplota).

Mehanička energija predstavlja zbir potencijalne i kinetičke energije u mehaničkom sastavu, odnosno energija koja zavisi od položaja i pomeranja tela usled delovanja sile. Hidro-pumpno skladištenje energije podrazumeva skladištenje u obliku gravitacione potencijalne energije vode. Izgled ovog tipa skladišta dat je na slici 2.



Slika 2. Šematski prikaz reverzibilne hidroelektrane

Glavna razlika između konvencionalnih i reverzibilnih hidroelektrana, je u tome što kod konvencionalnih voda iz akumulacionog jezera protiče kroz postrojenje i dalje nastavlja svojim prirodnim tokom, dok kod reverzibilnih postoje dva skladišta vodene mase, odnosno postoje gornji i donji rezervoar (akumulacija). Pored pomenutog hidro-pumpnog načina za skladištenje energije postoje i sledići: spremnici elastične energije, skladištenje energije komprimovanim vazduhom, spremnici kinetičke energije (zamajci).

Proizvedena toplota u nekom tehnološkom procesu, može biti višak toplote iz termoelektrane kao i neka otpadna toplota koju je šteta baciti u okolinu pa se skladišti, i privremeno se skladišti u spremnicima za toplotu na odgovarajućoj temeraturi kako bi se posle mogla koristiti. Uskladištena toplotna energija se kasnije može koristiti za zagrevanje potrošne tople vode i za grejanje prostorija ili za proizvodnju električne energije.

Sistem za skladištenje elekto-hemijske energije je zasnovan na principu skladištenja hemijske energije koja se pretvara u električnu. Ovde spadaju tri grupe uređaja: primarne baterije, sekunarne baterije i gorivne ćelije. Litijum-jonske baterije predstavljaju najbolje rešenje kada je u pitanju skladištenje velike količine obnovljive energije.

Superprovodno magnetno skladištenje energije (SMSE) karakterističan je po tome što ima tri principa, koja su samo za njega vezani, a to su: neki materijali provode struju bez termogenih gubitaka, električna struja stvara magnetno polje i magnetno polje je čist oblik energije koji može da se skladišti.

Shodno tome da je cilj da se snabdevanje energijom bazira na 100% obnovljivom energijom, problem se javlja prvo u skladištenju same električne energije, koja je znatno skuplja u odnosu na ostale tehnologije skladištenja, a onda u pretvaranju električne energije u različite skladišne i prenosive nosioce energije.

3.3. Pametni energetski sistemi

Definicija pametnih energetskih sistema je takva da se odnosi na kombinaciju električne, toplotne i gasne mreže u cilju pronalaženja najboljeg rešenja u celokupnom sistemu. Ovakvi sistemi zahtevaju nove tehnologije i infrastrukturu, koje stvaraju nove oblike fleksibilnosti, prvenstveno u fazi konverzije energetskog sistema. Prednost ovakvog tipa sistema se ogleda u tome da ne postoji zavisnost od varijabilnih obnovljivih izvora. Integracijom sektora grejanja i hlađenja sa električnom energijom, omogućava veću efikasnost goriva i povećanje udela varijabilnih resursa, što rezultira efikasnijim sistemom i rešenjima sa najnižim troškovima.

Studije za nekoliko pojedinačnih zemalja Evrope, došle su do zaključka da je najjeftiniji način snabdevanja grejanjem kombinovanje uštede toplote sa daljinskim grejanjem u urbanim područijima i pojedinačnim toplotnim pumpama u ruralnim područijima.

Ono što se smatra pod bitnom tačkom, jeste elektrifikacija transportnog sektora, jer bi se time osigurala ravnoteža između proizvodnje i potražnje u elektroenergetskom sistemu.

4. PRIMERI DOBRE PRAKSE

Primeri se pokazuju koje su sve predosti u primeni tehnolohije koja poseduje kogeneraciju i skladištenje energije.

4.1. Ušteda energije u zgradama: kogeneracija i kogeneracija zajedno sa skladištenjem toplotne energije

U ovom primeru dato je izvršeno istraživanje električne energije i rashladnog otperećenja u zgradama Azijskog tehnološkog instituta (AIT), Bangkok, Tajland. Zatim je izvedena studija izvodljivosti, kako tehnička tako i ekonomska, kogeneracije pomoću dvostrukog efekta apsorpcionog hladnjaka i upoređuje se kogeneraciona sprega sa termoenergetskim skladištem rashlađene vode.

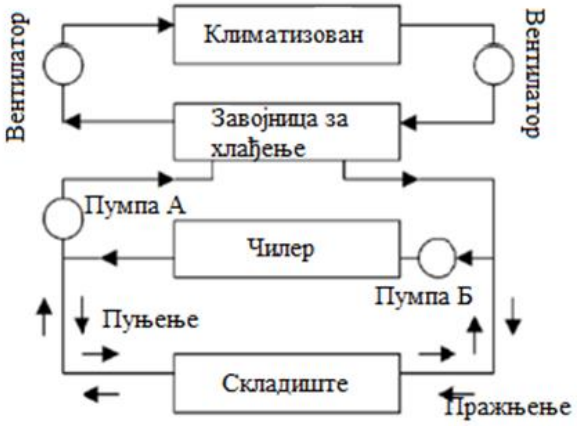
U radu se izučava da li je izvodljivo kogeneraciono postrojenje u cilju što bolje upotrebe energije u zgradi pomoću dvostrukog efekta apsorpcionog hladnjaka. Izabrano je delimično usklađivanje snage sa ciljem smanjenja računa za električnu energiju, kao i postizanja niskih troškova proizvodnje električne energije.

Količina rashladnog opterećenja proizvedenog kogeneracijom zavisi od performansi apsorpcionog hladnjaka sa dvostrukim efektom, koji varira u zavisnosti od količine zagrevanja u izduvnim gasovima i rashladnoj vodi omotača motora i njihovih temperatura.

Ukupna količina efekta hlađenja proizvedena za svaki scenario izračunata je prema rasporedu opterećenja motora. Računanje je izvedeno uzimajući u obzir vrednost (COP) od 0,8 za apsorpcioni hladnjak sa dvostrukim efektom. Uštede električne energije i potražnje izračunate su korišćenjem COP konvencionalnog sistema.

Primećeno je da kogenerirani rashladni kapacitet premašuje potrebe Instituta za rashladnim opterećenjem, osim tokom radnog vremena. To ukazuje na potrebu za skladištenjem ohlađene vode.

Sistem za skladištenje toplotne energije (TES) može se povezati sa sistemom za kogeneraciju kako bi se u potpunosti iskoristila kogenerativna ohlađena voda. Odabir najboljeg sistema za skladištenje rashlađene vode se vrši na osnovu klasifikacije po veličini i metode koja ima cilj da umanji mešanje ili prenos toplote između uskladištene vode za hlađenje i toplije vode koja se vraća iz sistema za hlađenje objekta, zgrade. U ovom radu je posmatran tip sistema sa delimičnim skladištenjem: on je hibrid delimičnog i potpunog načina skladištenja. Šema radnog ciklusa skladištenja rashlađene vode data je na slici 3.



Slika 3. Šema radnog ciklusa skladištenja rashlađene vode

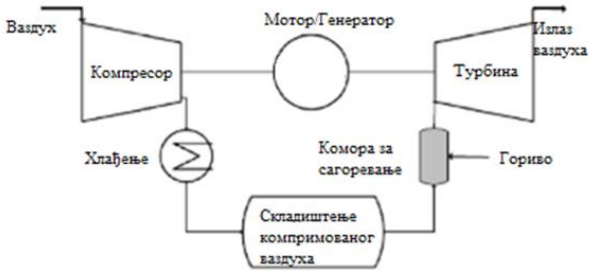
Tehnička procesa TES-a je izvedena određivanjem viška količine kogeneriranog rashladnog opterećenja dostupnog za skladištenje, potrebne veličine rezervoara za skladištenje i određivanjem odgovarajuće energije i potencijala uštede potražnje. U praksi se primenjuje da se skladišti više rashladne vode nego što je stvarno potrebno zbog gubitaka iz skladišta do kojih dolazi usled mešanja tokom punjenja i pražnjenja, kao i toplote iz okoline i sticanja iste tokom vraćanja toplote iz zgrade.

Na osnovu izvršenih istraživanja došlo se do rezultata koji pokazuju da je efikasnost sistema za skladištenje sa više rezervoara varirala od 85 do 98% u različitim zgradama. Ovaj proračun je izveden pretpostavljajući efikasnost skladištenja od 85%. Povezivanje kogeneracije sa TES ekonomičnije u poređenju sa samo kogeneracijom. Međutim, period povrata je uvek kraći kod kogeneracije spojene sa skladištem toplotne energije u poređenju sa onom samo kod kogeneracije.

4.2. Primene skladištenja energije komprimovanog vazduha u kogenerativnim sistemima

Skladištenje energije komprimovanog vazduha (CAES) predstavlja tehnologiju koja se može suprostaviti problemima koji nastaju prilikom skladištenja energije koja se dobija iz obnovljivih izvora energije, a koji se odnose na nestabilnost obnovljivih izvora pod uticajem prirode.

CAES sistemi se mogu podeliti na tri procsa: punjenje, skladištenje i pražnjenje, kao i na pet potprocesa: kompresija, punjenje i pražnjenje sa razmenom toplote, širenje, skladištenje i mehanička transmisija između motora/generatora, kompresora i ekspandera. Na slici 4. data je konfiguracija osnovnog CAES sistema.



Slika 4. Konfiguracija osnovnog CAES sistema

Jedna od klasifikacija odnosi se na proces prenosa toplote koji se odvija u procesima kompresije i ekspanzije: dijabatski-CAES (D-CAES), adijabatski-CAES (A-CAES) i izotermni-CAES (I-CAES). U radu su opisani tri najbitnija parametra za ocenu performansi: odnos primarne energije (PER), efikasnost povratnog putovanja (RTE) i efikasnost eksergije.

CAES sistemi samo skladište energiju, nisu sposobni da generišu različite vrste energije. Studija o ekonomskim i ekološkim efektima koji uključuju CAES u kombinovanoj termoelektrani (CHP) i obnovljivim izvorima sprovedena je međutim, nije se pokazala kao delotvorna.

CAES sistem kao kogeneracija sadrži nekoliko tipova. Trigenerativni CAES sistem T-CAES, ACCHP sistem koji koristi pneumatski motor kao ekspander, SC-CAES toplota od kompresije vazduha i hladnoća od širenja vazduha obnavljaju se kako bi se zadovoljili toplotni zahtevi u podhlađenom CAES, CCHP sistem zasnovan na plinskoj turbini kao i GT-CAES sistem, gde je glavna razlika između njih broj stepena kompresije. Pored navedenih postoji GT-S-CAES, CCHP sistem zasnovan na gasnoj turbini sa CAES integrisanom sa solarnim grejanjem, sistem kogeneracije zasnovan na dizelskom motoru i CAES (DE-CAES), zatim CCHP sistem zasnovan na gasnom motoru sa CAES (GE-CAES) i sistemom za ejektor za hlađenje.

Rezultati su pokazali da je kogeneracija sa CAES zaista središnje polje, pa je dalje teško sa samopouzdanjem identifikovati glavne trendove. Malo je očigledno da su mali GT-S-CAES-ORC sistemi najperspektivniji u pogledu efikasnosti eksergije, ali dalja ispitivanja koja to potvrđuju.

5. PRIMERI KOGENERACIJE I SKLADIŠTENJA ENERGIJE U SRBIJI

Srbija trenutno na svojoj teritoriji ne poseduje kogenertivno postojenje sa skladištenjem. Ono što je karakteristiše jesu nekoliko kogenerativnih postrojenja i jedno podzemno skladište gasa. Razvojem zemlje bi trebalo da se razvija i širi upotreba kogeneracije i skladištenja, upravo zbog njihovih doprinosa kako u očuvanju životne sredine tako i u pogledu efikasnosti koje oni daju.

Nekoliko kogenerativnih postrojenja u Srbiji su sledeća: kogenerativno postrojenje kompanije Imlek (Padinska Skela, Beograd), Biogasno postrojenje „Budućnost“ (Bačka Palanka), kogenerativno postrojenje EPS-a, kogenerativno postrojenje Voždovac i kogenerativno postrojenje pri Centru za upravljanje otpadom u Vinči. Dok što se tiče skladištenja, postoji samo jedno podzemno skladište gasa Banatski Dvor.

6. ZAKLJUČAK

U elektroenergetskom sistemu u svakom trenutku mora da postoji ravnoteža između energije koju proizvede određeno postrojenje, i energije koja je potrebna potrošačima. Od sada postoji mogućnost rešavanja ovog problema primernom opisane tehnologije, kogeneracije sa skladištenjem energije. Kombinacija određene tehnologije kogeneracije i tipa skladištenja energije, daje se velika mogućnost za različite privredne segmente.

Radom su obuhvaćene pomenute tehnologije ponaosob, ali i njihova primena zajedno na primerima dobre prakse. Prvi primer se odnosi na uštedu energije u zgradama: kogeneracija i kogeneracija zajedno sa skladištenjem toplotne energije, gde je izvršeno istraživanje električne energije i rashladnog otperećenja u zgradama Azijskog tehnološkog instituta (AIT), Bangkok, Tajland, dok se drugi primer odnosi na primenu skladištenja energije komprimovanog vazduha u kogenerativnim sistemima. Takođe, prikazan je uvid u primenu istih na teritoriji Srbije.

Cilj rada jeste da se ukaže na probleme koji nastaju u pogledu energije, čija potražnja raste iz godine u godinu. Upotrebom opisanih tehnologija bi se većinskim delom rešili problemi, jer bi se iskoristila energija koja se do sada baca i time doprinosi zagađenju, koje je sve veće kao i smanjenju efikasnoti postojećih postrojenja. Iz tog razloga, potrebno je osvestiti stanovništvo i probuditi svest o racionalnoj upotrebi energije. Datim primerima dat je uvid u troškove koji nastaju upotrebom samo jedne od tehnologija, a koji su troškovi kada se obe tehnologije koriste zajedno. Pored doprinosa u novčanom smislu, veliki uticaj njihove primene ima na povećanje efikasnosti postrojenja, čak i upotrebom fosilnih goriva.

6. LITERATURA

[1] Energy conservation in buildings: cogeneration and cogeneration coupled with thermal energy storage, https://www.sciencedirect.com/science/article/pii/S0306261903001004

[2] Applications of compressed air energy storage in cogeneration systems,

https://www.sciencedirect.com/science/article/pii/S0360544220320119

[3] Catalog of CHP Technologies,

https://www.epa.gov/chp/catalog-chp-technologies

[4] Održivo planiranje energije: tehnologije i energetska efikasnost,

https://www.researchgate.net/publication/327201030_13Poglavlje7_Skladistenje_energije

Kratka biografija:

Zorana Lazić rođena u Sremskoj Mitrovici 1998. god. Osnovne studije na Fakultetu tehničkih nauka iz oblasti Energetika i procesna tehnika -Termoenergetika završila je 2020. god. Trenutno student master studija na smeru Termoenergetika.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.7/.9
DOI: https://doi.org/10.24867/15AM04Djuranovic

ALUMINOTERMIJSKO ZAVARIVANJE ŽELEZNIČKIH ŠINA UZ PRIMENU INOKULANATA

THERMITE RAIL WELDING WITH INOCULANTS

Ilija Đuranović, Sebastian Baloš, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj- *U radu je izvršeno aluminotermijsko zavarivanje šina uz primenu i bez inokulanata. U okviru labaratorijskih ispitivanja sprovedena je ultrazvučna defektoskopija, metalografsko ispitivanje i izvedeno je ispitivanje tvrdoće zavarenog spoja. Kod sva tri uzorka registrovana je veća tvrdoća metala šava u odnosu na kontrolni uzorak. Generalno posmatrano u području metala šava najveću tvrdoću zabeležio je uzorak broj 1 koji sadrži najmanju količinu dodatih nano čestica titan dioksida TiO_2 u iznosu od 0,72 g. Osnovni razlog leži u modifikaciji mikrostrukture u smislu nestanka intragranularnog idiomorfnog ferita u modifikovanim uzorcima.*

Ključne reči: *aluminotermijski postupak, nano čestice, karakterizacija materijala*

Abstract- *In this paper, thermite welding of rails with and without inoculants was performed. Ultrasonic defectoscopy, metallographic analysis and hardness testing of the welds were performed. In all three samples, higher weld metal hardness was measured compared to the control sample. Generally, in the area of weld metal, the highest hardness was recorded in sample number one, which contains the smallest amount of added titanium dioxide TiO_2 nanoparticles of 0,72 g. The main reason for this is the modification of microstructure related to the absence of intragranular idiomorphic ferrite in modified specimens.*

Keywords: *thermite welding, nanoparticles, materials characterization*

1. UVOD

Zavarivanje predstavlja tehnološki postupak formiranja neraskidivih spojeva, koji su izvedeni uspostavljanjem međuatomskih veza između elemenata uz postojanje procesa difuzije, gde je neprekidnost strukture karakteristika takvih spojeva [1].

Metoda aluminotermijskog zavarivanja je postupak zavarivanja topljenjem, gde se koristi mešavina aluminijumskog praha i oksida gvožđa koji se na visokoj temperaturi, na oko 2500°C pretvaraju u aluminijum oksid i železo, odnosno čelik, ukoliko ima prisutnog ugljenika i legirajućih elemenata [2].

Sam termit predstavlja smešu aluminijumskih granula i metalnog oksida. Nakon što se sprovede početno paljenje, termit oslobađa jaku egzotermnu reakciju.

NAPOMENA:
Ovaj rad je proistekao iz diplomskog rada čiji mentor je bio prof. dr Sebastian Baloš.

Velika količina toplote, koja se oslobađa usled reakcije, omogućava dobijanje tečnog metala, bez korišćenja spoljašnjih izvora energije [3, 4].

U sastav termitne smeše ulazi smeša aluminijuma, ferolegura i oksida gvožđa, gde je cilj da sastav smeše bude podešen na taj način da se nakon završetka procesa zavarivanja sastav metala zavara što manje razlikuje od sastava osnovne šine [5].

Spajanje šina kod ovog procesa obezbeđuje se pomoću rastopa, koji ispunjava kalupnu šupljinu prethodno centriranog kalupa. Tečni čelik nastaje jakom egzotermnom hemijskom reakcijom između oksida železa i aluminijuma, koji ima ulogu redukcionog sredstva [5].

Prednosti procesa uključuju relativnu jednostavnost i brzinu ugradnje, fleksibilnost postupka za zavarivanje svih profila i vrsta šina i ekonomičnost.

Glavni nedostaci ovog postupka su mnogi koraci koje zavarivač u toku procesa rada mora primeniti, kao i uslovi okoline koji mogu rezultirati lošijim performansama zavarivanja. Ove promenljive uključuju stvari kao što su poravnanje krajeva šina, razmak na kraju šine, kvalitet pakovanja kalupa i porcije, trajanje predgrevanja, predgrevanje lončića i preusmeravajućeg čepa, vlaga u vazduhu i kristalizacije mesta zavara [6].

Najpoznatije metode naprednih postupaka aluminotermijskog zavarivanja poput SkV, SoW -5, THR, HPW, SRZ i SKS razvijene su od strane Elektro-termit Gmbh&Co, jednog od vodećih svetskih proizvođača opreme iz ove oblasti [3].

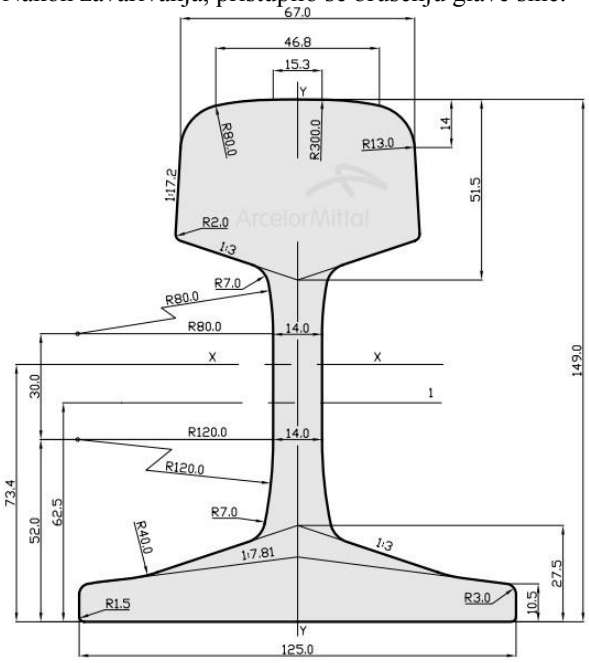
U okviru izrade master rada izvršeno je eksperimentalno aluminotermijsko zavarivanje šina, pri čemu je korišćen SOW postupak, bez i sa dodatim nanočesticama, u cilju utvrđivanja njihovog uticaja na osobine dobijenog zavarenog spoja.

2.EKSPERIMENTALNI RAD

2.1 Priprema uzoraka

Osnovni materijal je šina oznake 49E1, prikazana na Slici 1. Nominalni hemijski sastav klase materijala R260 je dat u tabeli 1. Za potrebe ovog eksperimentalnog rada aluminotermijskim SOW postupkom zavarivanja dobijeno je ukupno četiri uzorka, gde jedan uzorak predstavlja kontrolni uzorak 0 i prilikom njegove izrade nisu korišćene nanočestice, dok su preostala tri uzorka formirana uz dodatak nanočestica titan dioksida TiO_2. Uzorak 1 sadrži 0,72 g dodatih nanočestica titan dioksida TiO_2, uzorak 2 sadrži 2,82 g nano čestica, dok treći uzorak sadrži 4,98 g nanočestica. Količina nanočestica je

tako odmerena da je sadržaj nanočestica u glavi šine 0,1; 0,4 i 0,7 %. Nano čestice su bile postavljene u aliminijumsku foliju debljine 0,01 mm (10 μm), mase 0,15; 0,22 i 0,44 g, koja je vezana bakarnom žicom za keramički čep sa gornje strane kalupa, tako da se nalazi u prostoru koji odgovara mestu gde se nalazi glava šine. Nakon zavarivanja, pristupilo se brušenju glave šine.



Slika 1 *Korišćena šina 49E1 (S49) sa dimenzijama profila [7]*

Tabela 1 *Hemijski sastav prema EN13674-1:2011, ostatak Fe, u [masenim %]*

C	**Mn**	**Si**	**P**	**S**	**Ni**
0,62-0,8	0,7-1,2	0,15-0,58	do 0,025	do 0,025	do 0,10
Mo	**Al**	**Cr**	**V**	**Cu**	**Nb**
do 0,02	do 0,02	do 0,15	do 0,13	do 0,09	do 0,01

Pre nego što se pristupilo ultrazvučnom ispitivanju šinskih profila, šinski profili su čišćeni krpom i žičanom četkom, kako bi se uklonile masti i grube nečistoće. Nakon toga pre naslanjanje sondi na šinski profil i pre početka ultrazvučne defektoskopije, bilo je neophodno premazati ispitivanu površinu lubrikantom, odnosno ultrazvučnim gelom. Nakon toga uzorak je bio spreman za početak ultrazvučnog ispitivanja.

Nakon ultrazvučnog ispitivanja, isečeni su uzorci dimenzija 10x10x150 mm trakastom testerom sa vrha šine za metalografsku analizu i merenje tvrdoće. Metalografska priprema se sastojala od montiranja, brušenja, poliranja i nagrizanja nitalom (3 % azotne kiseline HNO_3 u alkoholu).

2.2 Ultrazvučna ispitivanja

Primenjena ultrazvučna defektoskopija imala je za cilj otkrivanje defekata, odnosno grešaka u materijalu šine, što se vrši ispitivanjem zapremine zavarenog spoja i zone uticaja toplote. Prilikom ultrazvučnog ispitivanja kontrolnog uzorka, kao i uzoraka sa dodatim nanočesticama titan dioksida TiO_2 korišćen je ultrazvučni uređaj tipa UCD-50 IPS. Dozvoljena veličina greške je 5 mm.

2.3 Metalografska ispitivanja materijala

Metalografija predstavlja nauku koja se bavi ispitivanjem mikrostrukture, odnosno unutrašnje građe i osobina metala i legura. Mikroskopskim ispitivanjem dobija se potpunija slika o unutrašnjoj građi materijala. Ovakav vid ispitivanja pruža mogućnost da se odredi tip mikrostrukture, udeo određenih mikrokonstituenata, veličina zrna, njihova raspodela, orijentacija, kao i veličina uključaka.

Greške koje narušavaju homogenost strukture, a mogu nastati primenom različitih tehnoloških procesa, mogu se ustanoviti ovim vidom ispitivanja [8]. Uzorci su ispitivani na svetlosnom mikroskopu Leitz Ortoplan.

2.4 Ispitivanje tvrdoće

Tvrdoća metala i legura predstavlja otpor koji taj materijal pruža prilikom prodiranja nekog tvrđeg tela. Tvrdoća je indikator čvrstoće materijala, ali se ujedno koristi kao indikator pojave zakaljenja u zoni uticaja toplote. U radu je korišćena metoda po Vikersu, na istim uzorcima koji su korišćeni za metalografska ispitivanja.

Izvršena su po tri merenja tvrdoće sa obe strane osnovnog materijala (OM), u zoni uticaja toplote (ZUT) i u metalu šava (MŠ), na tri mesta: na sredini tri merenja i po tri merenja prema ZUT-u (MŠ1 i 2).

3.REZULTATI I DISKUSIJA

3.1 Ultrazvučna defektoskopija

Defektoskop, odnosno sonda registrovala je u zavarenom spoju i oko zavarenog spoja (ZUT) minimalne ehoe i šumove, što je znak da nema značajnijih grešaka i nepravilnosti koje bi na bilo kakav način mogle ugroziti postojanost zavarenog spoja.

Mali pikovi, odnosno šumovi koji su zabeleženi na defektoskopu pokazuju da je mikrostruktura povoljna sa stanovišta postojanja grešaka. Pikovi koji se pojavljuju na ovaj način, produkt su postojanja sitnih grešaka u vidu prslina, gasnih pora i drugih uključaka ili su odraz odbitaka ultrazvučnog talasa od neke geometrije zavarenog spoja.

Pojačanje kao izlazni parametar je pokazatelj koji govori, gde se prilikom ispitivanja lakše probija zvučni zid. Kod kontrolnog uzorka 0 i uzorka 1 registrovano je veće pojačanje na mestu zavarenog spoja nego na osnovnom materijalu, dok je kod uzorka 2 i 3 obrnuta situacija.

Sve u svemu, svi ispitani uzorci zadovoljavaju kriterijume koji su postavljeni pred zavarene spojeve železničkih šina dobijene aluminotermijskim postupkom, pre svega najveću dozvoljenu veličinu defekta (pora ili nemetalni uključak) od 5 mm.

Tabela 2 *Prikaz vrednosti izlaznog parametra pojačanja u području osnovnog materijala*

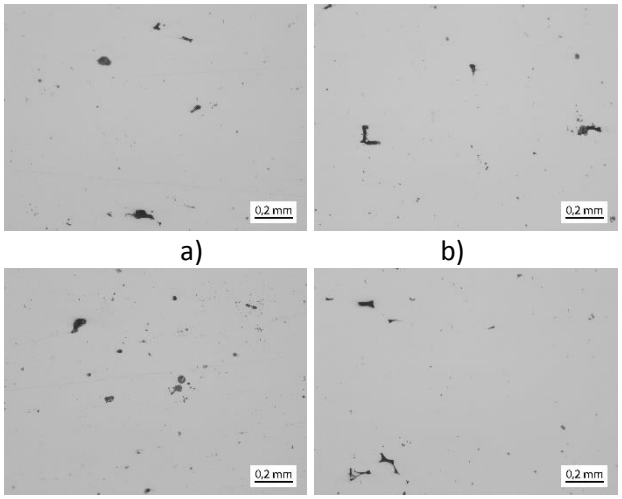
	Pojačanje	**Jedinica mere**	**Područje uzorka**
Kontrolni uzorak 0	25	dB	Osnovni materijal
Uzorak 3	25	dB	Osnovni materijal

Tabela 3 *Prikaz vrednosti izlaznog parametra pojačanja u području metala šava*

Redni broj uzorka	Pojačanje	Jedinica mere	Područje uzorka
Kontrolni uzorak 0	27	dB	Metal šava
Uzorak 3	24	dB	Metal šava

3.2 Metalografska ispitivanja

Izgled poliranih uzoraka prikazan je na slici 2. Kod svih uzoraka se vidi prisustvo uključaka u čvrstom stanju (nemetalnih uključaka) i gasnih uključaka (pora). Ove nepravilnosti su dozvoljene i uobičajene su kod zavarenih spojeva relativno velikih poprečnih preseka kao što su spojevi aluminotermijskim postupkom. Ovakvi rezultati su u skladu sa rezultatima ultrazvučnog ispitivanja.



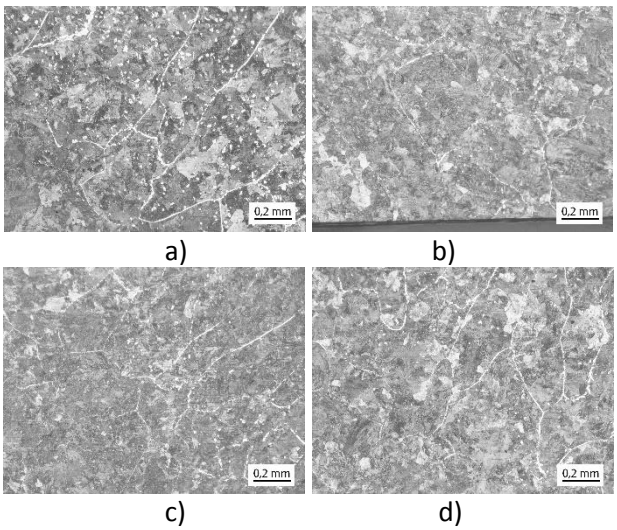
Slika 2 *Polirani uzorci: a) kontrolni uzorak 0, b) uzorak 1, c) uzorak 2, d) uzorak 3*

Dobijene mikrostrukture u metalu šava pokazali su dominantno prisustvo perlita (tamna faza) uz manju količinu ferita (svetla faza). Ferit je kod svih uzoraka raspoređen oko prvobitnih austenitnih zrna u vidu dugih svetlih linija (alotriomorfni ferit-AF) kao i unutar prvobitnih austenitnih zrna (intragranularni idiomorfni ferit-IIF), slike 3 i 4. Međutim, u blizini linije stapanja, unutar metala šava (slika 2), sadržaj obe vrste ferita je manji kod uzoraka sa nano česticama, naročito IIF.

Osim toga, vidi se da su prvobitna austenitna zrna, čije se konture vide na osnovu AF izdužena i da stvaraju tipičnu stubastu mikrostrukturu koja se javlja u metalu šava zavarenih spojeva.

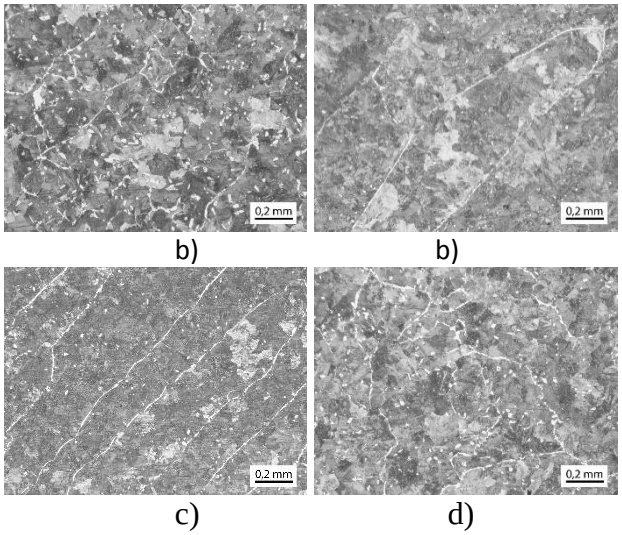
Pravac ovih kristalita je jednak pravcu odvođenja toplote, ali je njihov rast u suprotnom smeru od smera odvođenja toplote.

U centralnom delu metala šava, IIF kod uzoraka zavarenih uz prisustvo nano čestica prisutan je u većoj količini nego u zoni prema liniji stapanja, ali je još uvek niži u odnosu na uzorak 0.



Slika 3 *Mikrostruktura prelazne zone; dole levo je ZUT, ostatak je metal šava: a) kontrolni uzorak 0, b) uzorak 1, c) uzorak 2, d) uzorak 3*

Mikrostruktura ZUT-a je nepromenjena kod svih uzoraka, a isto važi i za osnovni materijal.



Slika 4 *Mikrostruktura centralnog dela metala šava: a) kontrolni uzorak 0, b) uzorak 1, c) uzorak 2, d) uzorak 3*

3.3 Ispitivanje tvrdoće

Na osnovu dobijenih rezultata srednjih vrednosti tvrdoće može se konstatovati da je tvrdoća modifikovanih uzoraka u području metala šava značajno veća u odnosu na nemodifikovani uzorak. Ovo povećanje je proporcionalno veće nego što je uzorak osnovnog materijala tvrđi u odnosu na nemodifikovani materijal, dok su vrednosti u ZUT-u slične, tabela 4.

Najveće srazmerno povećanje tvrdoće je u zoni metala šava pored linije stapanja (MŠ1 i MŠ2), upravo na mestu gde je primećeno smanjenje sadržaja IIF pri metalografskim ispitivanjem, slika 3. Najveće povećanje tvrdoće izmereno je na uzorku 1, sa najmanjim sadržajem nano čestica, gde je i sadržaj IIM najmanji.

Tabela 4 *Srednje vrednosti tvrdoće [HV] dobijenih rezultata ispitivanja*

Redni broj uzorka	OM	ZUT	MŠ-1	MŠ	MŠ-2	ZUT	OM
0	252	294	255	268	268	310	254
1	255	309	302	288	301	295	258
2	280	306	305	280	280	301	253
3	266	291	296	278	299	315	293

Sa gledišta mehaničkih osobina, svi modifikovani uzorci (1, 2, 3) imaju bolje mehaničke osobine u odnosu na uzorak 0. Optimalni uzorak je uzorak 1, dobijen sa dodatim nanočesticama, a osnovni razlog je smanjen sadržaj ferita. S obzirom da ferit nastaje pre perlita, može se pretpostaviti da je prisustvo nano čestica uticalo na ograničavanje rasta ferita, tako da je u blizini linije topljenja došlo do gotovo potpunog zaustavljanja nastanka IIF kod uzoraka sa nano česticama.

Umesto nastanka IIF i AF, nastaje perlit. Za detaljnije objašnjavanje uticajnih fenomena korisno je nastaviti ispitivanja u pogledu zatezanja, savijanja i energije udara. Najveći efekat je prisutan kod uzorka sa najmanje dodatih nano čestica, verovatno kao posledica aglomeracije kod uzoraka sa više dodatih nano čestica.

5. ZAKLJUČCI

Na osnovu dobijenih rezultata, mogu se izvući sledeći zaključci:

- Dodavanje nanočestica nema negativan uticaj sa gledišta ultrazvučnog ispitivanja.
- Kao i kod kontrolnog uzorka, kod modifikovanih uzoraka u metalu šava nastaju pore i nemetalni uključci
- Mikrostruktura metala šava u zoni blizu linije topljenja ima manji sadržaj ferita, kako idiomorfnog intragranularnog, tako i alotriomorfnog.
- Tvrdoća u svim zonama je povećana, a najviše u metalu šava pored linije topljenja, što je posledica smanjenog sadržaja ferita.
- Najverovatniji uzrok nestanka ili smanjenja sadržaja ferita iz pojedinih zona jeste ograničavanje rasta usled prisustva nanočestica ili čestica, verovatno oksida.
- Najveći stepen ojačanja je dobijen kod uzorka sa najmanje dodatih nano čestica, najverovatnije zbog pojave aglomeracije tj. spajanja nano čestica kod uzoraka sa većim dodatkom nano čestica.

Na osnovu iznetog, može se zaključiti da unos nano čestica može da bude efikasan način ojačavanja aluminotermijskih zavarenih spojeva.

6. ZAHVALNOST

Rezultati prezentovani u ovom radu su realizovani u okviru projekta „Inovativni materijali i tehnologije spajanja“, Departmana za proizvodno mašinstvo, FTN, Novi Sad.

7. LITERATURA

[1] V. Palić, „*Zavarivanje*“, Novi Sad: Fakultet tehničkih nauka, 1987.
[2] M.J.M.M.Steenbergen, R.W. Van Bezooijen, „*Rail welds, Delfi University of Technology*“, Id Consultancy, Netherlands, 2017.
[3] https://www.elektro-thermit.de/en/rail-joining/thermitr-welding-processes/ (Pristupio – 18.09.2021.)
[4] http://www.railsystem.net/thermit-welding/ (Pristupio – 20.06.2021.)
[5] D. Ilić, „*Aluminotermijsko zavarivanje tračnica*“, Završni rad, Sveučilište u Zagrebu fakultet strojarstva i brodogradnje, 2015.
[6] C.P. Londsdale, „*Thermite rail welding: History, process developments, current practices and outlook for the 21st century*“, Metallurgical Engineer Conrail Technical Services Labaratory, Altoona, USA, 2000.
[7] https://rails.arcelormittal.com/types-rails/transport-rails/european-standards/rail-s49-49e1 (Pristupio 21.09.2021)
[8] https://vts.edu.rs/wp-content/uploads/2017/11/4-TEHNOLO%C5%A0KA-ISPITIVANJA.pdf (Pristupio – 16.09.2021)

Kratka biografija:

Ilija Đuranović rođen je u Novom sadu 1996. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Proizvodno mašinstvo – Tehnologije spajanja odbranio je 2019.god.

Sebastian Baloš rođen je u Somboru 1974. god. Doktorirao je na Fakultetu tehničkih nauka 2010. god., a 2021. godine je biran u zvanje redovnog profesora iz oblasti Mašinsko inženjerstvo, tj. uže naučne oblasti Materijali i tehnologije spajanja.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/15BE01Radonjic

SISTEM ZA ANALIZU MAMOGRAFIJE DOJKE POMOĆU RAČUNARA

COMPUTER AIDED DIAGNOSIS SYSTEM FOR ANALYZING BREAST MAMMOGRAPHY

Stefan Radonjić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – *Cilj ovog rada ogleda se u razvoju grafičkog-korisničkog interfejsa (engl. Graphical User Interface – GUI), baziranog na algoritmima iz tradicionalne računarske vizije, koji će potencijalno olakšati intelektualni napor i redukovati utrošeno vreme radiologa prilikom analize i intereptacije mamografije dojke u kliničkoj praksi.*

Ključne reči: *karcinom, mamografija, analiza slike, obrada slike, grafičko korisnički interfejs, mašinsko učenje.*

Abstract - *The aim of this paper is to develop a graphical user interface (GUI), based on algorithms from traditional computer vision, which will potentially facilitate reduction of the intellectual effort as well as reduce the time spent by radiologists when analyzing and interpreting breast mammography in clinical practice.*

Key Words - *cancer, mammography, image analysis, image processing, graphical user interface, machine learning.*

1. UVOD

Snimanje, kao važan deo kliničkih protokola za uspostavljanje kontrole nad karcinomom može pružiti razne informacije o morfologiji, strukturi, metabolizmu i funkcijama karcinoma. Različite tehnike snimanja mogu pružiti dodatne informacije koje se koriste za poboljšanje planiranja terapije pacijenta.

Glavna svrha snimanja ogleda se u pronalaženju minimalno invazivne terapije radi postizanja boljih rezultata i smanjivanja neželjenih efekata. Jedan od najvažnijih faktora za smanjene smrtnosti određenih vrsta karcinoma ogleda se u uspostavljanju rane dijagnoze karcinoma kod pacijenata putem snimanja i interpretacije slika organa od interesa.

Najčešća vrsta karcinoma kod žena jeste karcinom dojke koji se smatra drugim vodećim uzrokom smrti od karcinoma kod žena. Snimanje dojke je uvek predstavljalo deo nege karcinoma dojke i koristilo se u svim fazama lečenja karcinoma, od otkrivanja i postavljanja do praćenja terapije i post-terapeutskog praćenja. Opšti pojam snimanja dojke odnosi se na sonografiju dojke, mamografiju i magnetnu rezonancu tomografiju (MRT) dojke.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Milan Segedinac, vanr. prof.

Shodno tome, prvi deo rada, naslovljen teorijske osnove, daje pregled oblasti analize digitalnih slika. Drugi deo, pregled različitih algoritama za obradu digitale slike podržavanih od strae sistema. Treći deo rada purža kratak opis grafičkog korisničkog interfejsa (arhitekture sistema) i ilustruje funkcionalnosti specificirane iz prethodnog odeljka.

2. TEORIJSKE OSNOVE

2.1. Analiza digitalnih slika

Polje za analizu digitalnih slika omogućava računarima da izdvoje, modifikuju, poboljšaju i olakšaju ljudima proces interpretacije digitalne slike. Iz prethodno navedeih razloga, oblast analize digitalnih slika ima dubok uticaj na mnoge druge oblasti, u rasponu od astronomije do nanotehnologije. Razvoj novih uređaja za obradu i analizu digitalnih slika revolucionisao je oblast medicinskih nauka. Međutim, podaci (slike/video zapisi) prikupljeni ovim uređajima često se tumače vizuelno, što je dosadna, psihički zamorna praksa i sklona greškama.

Analiza digitalnih slika pruža skup metoda i alata koji pomažu biolozima/kliničarima da uspostave preciznu dijagnozu i donesu zaključke o različitim medicinskim stanjima. Filtriranje digitalnih slika odnosi se na modifikovanje intenziteta piksela slike na osnovu neke unapred definisane funkcije lokalnog komšiluka. Filtriranje se često smatra preduslovom za većinu zadataka vezanih za analizu slike.

Glavni cilj poboljšavanja slike jeste da se slika obradi na takav način da rezultat obrade bude pogodniji od orginale slike za specifičnu aplikaciju. Dakle, metoda koja je vrlo pogodna za poboljšavanje rendgenskih slika ne mora nužno biti korisna za poboljšavanja *CT* slika. Slično tome, metoda korištena za poboljšavaje magnetne rezonance mozga, ne mora nužno biti korisna za poboljšavanje magnetne rezonance nekog drugog organa.

Pristupi za poboljšavanje slike se mogu grubo podeliti u dve široke kategorije:

1. Metode za obradu slike u prostornom domenu, čije su tehnike bazirane na direktnoj manipulaciji intenziteta piksela slike.
2. Metoda za obradu slike u domenu frekvencije čije su tehnike bazirane na modifikaciji Furijeove transformacije slike.

Metode prostornog domena su procedure koje direktno barataju sa pikselima slike. Procesi izvršavani u okviru prostornog domena će se biti označavani sledećim izrazom:

$$g(x,y) = T[f(x,y)]$$

gde je $f(x,y)$ ulazna slika, $g(x,y)$ je obrađena slika, a T je operator koji se izvršava nad f, i koji je definisan nad nekim komšilukom datog piksela (x,y). Takođe, operator T može da se izvršava i na skupu ulaznih slika, kao što je na primer izvođenje sume korespodentnih piksela K-slika u cilju redukovanja šuma.

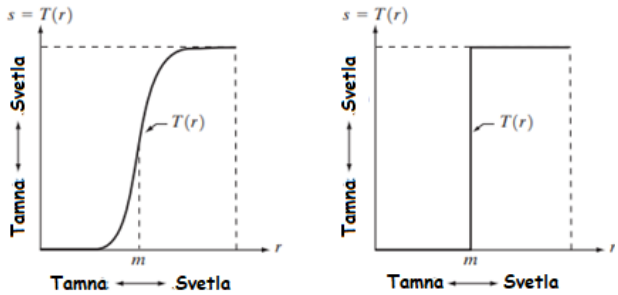
Glavni pristup prilikom definisanja komšiluka tačke (x,y) jeste upotreba kvadratne ili pravougaone površine, odnosno podslike sa centrom u tački (x,y). Centar podslike se pomera sa piksela na piksel, počevši, recimo, od gornjeg levog ugla, s leva na desno, pa tako sve do donjeg desnog ugla. Operator T se primenjuje na svakoj (x,y) lokaciji kako bi se dobio izlaz, g, na istoj lokaciji. Proces koristi samo piksele u području slike kojim se prostire komšiluk

Najjednostavniji oblik operatora T je kada postoji samo 1 sused, odnosno, kada je matrica suseda dimenzija $1x1$. U ovom slučaju, izlaz, g, zavisi isključivo od vrednosti f na (x,y) poziciji, a operator T postaje funkcija transformacije inteziteta koja se formalno može zapisati u sledećem obliku:

$$s = T(r)$$

gde, radi jednostavnosti notacije, r i s predstavljaju promenljive, koje redom označavaju, nijansu sive ulazne slike $f(x,y)$ i procesuirane slike $g(x,y)$ na bilo kojoj (x,y) koordinati. Na primer, ako $T(r)$ ima oblik prikazan na slici 2.1. (a), efekat ove transformacije bio bi stvaranje slike sa većim kontrastom od originalne, zatamnjivanjem intenziteta/nijansi piksela ispod nivoa m i posvetljivanjem intenziteta/nijansi piksela iznad nivoa m na originalnoj slici.

U okviru prethodno opisane tehnike, koja je u literaturi takođe poznata i pod nazivom „*protezanje kontrasta*“, vrednosti promenljive r, odnosno, intenzitet piksela na (x,y) poziciji originalne slike, koje su manje od unapred definisane margine (nijanse piksela), m, se komprimuju pomoću funkcije za transformaciju intenziteta piksela, T, u skladu sa uskim opsegom promenljive s, odnosno, željenog intenziteta izlaza/rezultata obrade, prema crnoj boji. Suprotan efekat se dešava za vrednost intenziteta piksela r iznad margine m. U slučaju prikazanom na slici 2.1. (b), $T(r)$ generiše binarnu sliku. Mapiranje ovog oblika, T, se takođe naziva i funkcijom praga (*engl. thresholding fuction*).



Slika 2.1. *funkcija mapiranja/transformacije nijansi sive za poboljšavanje kontrasta slike. Sa leve strane slike je prikazana funkcija za istezanje kontrasta (engl. Contrast Stretching), dok je s desne strane slike prikazana funkcija za proizvodnju binarne-slike (engl. Thresholding Function).*

Veće susedstvo nam automatski pruža i veći stepen fleksibilnosti. Opšti pristup je da se koristi funkcija koja na ulazu očekuje vrednost, odnosno intenzitet piksela ulazne slike f, u unapred definisanom susedstvu piksela na (x,y) poziciji, za određivanje vrednosti g na istoj (x,y) poziciji. Jedan od glavnih pristupa u ovoj formulaciji zasnovan je na upotrebi takozvanih maski , koje se nazivaju i filterima, šablonima, jezgrima (*engl. kernel*) ili prozorima. U osnovi, maska je mali (recimo, 3x3) 2-D niz u kojem vrednosti koeficijenata maske određuju prirodu procesa, kao što je na primer pooštravanje slike. Tehnike poboljšavanja slike zasnovane na ovoj vrsti pristupa često se nazivaju filtriranjem (*engl. filtering*).

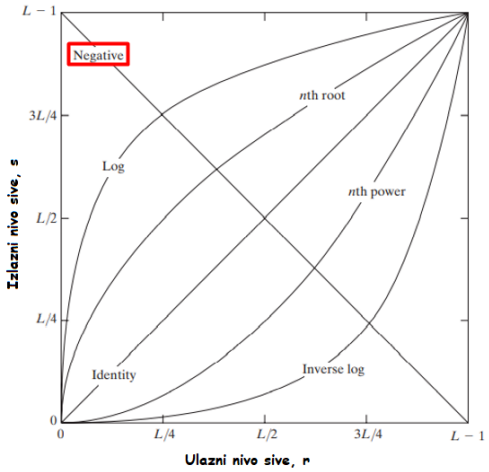
3. FUNKCIONALNOSTI PODRŽAVANE OD STRANE SISTEMA

3.1. Negativ slike

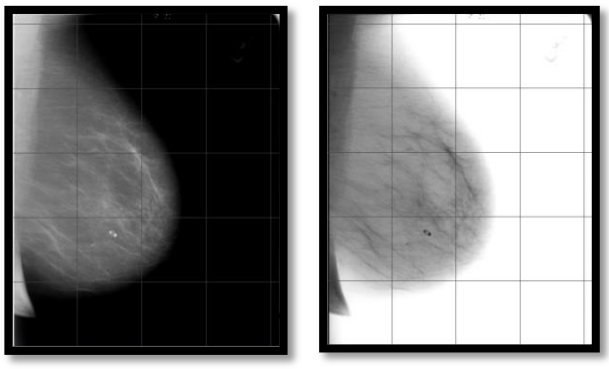
Negativ slike čiji se intenzitet, odnosno nijanse sive nalazi u rasponu $[0, L-1]$ dobija se uz pomoć negativne trasformacije prikazane na slici 3.1, koja je data izrazom:

$$s = L - 1 - r$$

Obrtanje nivoa intenziteta slike na ovaj način daje ekvivalent fotografskom negativu. Ovakva vrsta obrade je posebno pogodna za poboljšavanje belih ili sivih detalja ugrađenih u tamne delove slike, posebno kada su crna područja dominantna u veličini. Primer je prikazan na slici 3.2. Originala slika je digitalni mamograf koji prikazuje malu leziju. Uprkos činjenici da je vizuelni sadržaj na obe slike isti, uočiti koliko je u ovom slučaju lakše analizirati tkivo dojke na negativnoj slici.



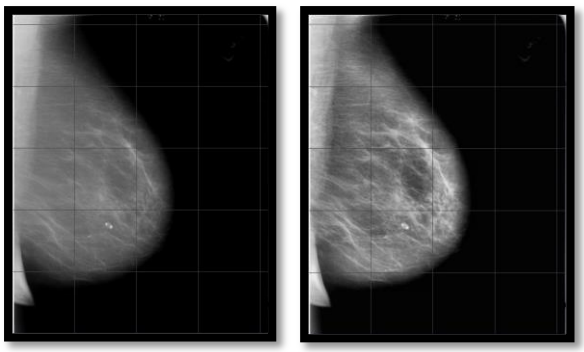
Slika 3.1. *Neke od osnovnih tipova funkcija baziranih na transformaciji nijansi sive za poboljšanje slike. Slika je preuzeta iz [1]*



Slika 3.2 *(a) originalni digitalni mamograf (b) Negativna slika dobijena korišćenjem negativne transformacije T(r) = L – 1 – r*

3.2. Adaptivno izjednačavanje histograma sa ograničenim kontrastom (CLAHE)

Adaptivno izjednačavanje histograma (AHE) je računarska tehnika obrade slika koja se koristi za poboljšanje kontrasta na slikama. Razlikuje se od običnog izjednačavanja histograma po tome što adaptivni metod izračunava nekoliko histograma, od kojih svaki odgovara određenom delu slike, i koristi ih za preraspodelu vrednosti lakoće slike. Zbog toga je pogodan za poboljšanje lokalnog kontrasta i poboljšanje definicija ivica u svakom delu slike. Međutim, AHE ima tendenciju da pojačava šum u relativno homogenim delovima slike. Varijanta adaptivnog izjednačavanja histograma nazvana adaptivno izjednačavanje histograma ograničeno kontrastom (CLAHE) [2] sprečava ovo ograničavanjem pojačanja, vidi sliku 3.4.



Slika 3.4. *Na levoj strani je prikazana originalna slika mamograma dojke, a na desnoj slika nakon adaptivnog izjednačavanja histograma ograničenog kontrastom.*

3.3. Prilagodljiva gama korekcija sa raspodelom pondera (AGCWD)

Generalno govoreći, funkcija gustine verovatnoće (*engl. Probability Density Function -PDF*) i funkcija kumulativne raspodele (*engl. Cumulative Distribution Function – CDF*) mogu se koristiti za poboljšanje intenziteta piksela, iako literatura ukazuje na to da se osvetljenost u okviru slike može izobličiti [14] - [16]. S druge strane, tradicionalni metod gama korekcije koristi funkciju konstantnog stepenovanja sa eksponentom γ za poboljšavanje slike. Glavna mana ove metode leži u potrebi za manuelnim podešavanjem vrednost parametara gama. Inspirisani teorijom verovatnoće i statističke inferencije, autori [3] predlažu utvrđivanje adekvatne vrednosti γ parametra uz pomoć sledeće transformacije:

$$T(l) = 255\left(\frac{1}{255}\right)^{1-CDF(l)}$$

gde $l = l_{min},\ l_{min}+1, l_{min}+2, \dots, l_{max}$. Nažalost, CDF kriva prigušene slike doživljava značajne fluktuacije usled pojava u okruženju, prema prethodnim istraživanjima [4] - [6]. Kao rezultat, prethodna jednačina može stvoriti nepovoljne artefakte. Da bi rešili ovaj problem, autori [7] koriste funkciju raspodele pondera (*engl. weighting distribution function*) [7] za izravnavanje fluktuirajuće pojave. Funkcija raspodele pondera može se izraziti na sledeći način:

$$PDF_w(l) = PDF_{max}\left(\frac{PDF(l) - PDF_{min}}{PDF_{max} - PDF_{min}}\right)^{\alpha},$$

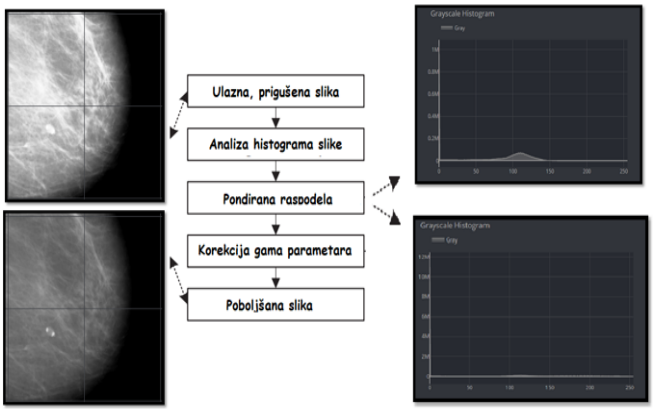
gde $l = l_{min},\ l_{min}+1, l_{min}+2, \dots, l_{max}$, $PDF_w(l)$ predstavlja gustinu raspodele verovatnoće pondera (engl. weighting probability distribution), PDF_{max} predstavlja maksimalnu gustinu verovatnoće, PDF_{min} predstavlja minimalnu gustinu verovatnoće, a α predstavlja adaptivi parametar. Koristeći PDF_w, orginalna CDF funkcija se izravnava i može biti predstavljena kao:

$$CDF_s(l) = \sum_{h=0}^{l} \frac{PDF_w(h)}{\sum PDF_w},$$

gde $l = l_{min},\ l_{min}+1, l_{min}+2, \dots, l_{max}$, $\sum PDF_w$ predstavlja sumu linearne kombinacije verovatnoća, i $CDF_s(l)$ predstavlja izravnatu CDF funkciju. Konačno, utvrđivanje adekvatne vrednosti γ parametra može se modifikovati i zapisati u sledećem obliku:

$$T(l) = 255\left(\frac{1}{255}\right)^{1-CDF_s(l)}$$

Slika 3.5 prikazuje dijagram toka predložene metode za poboljšavanje kontrasta slike. Za ulaznu prigušenu, odnosno zatamljenu sliku, većina piskela je gusto raspoređena u području tkiva parenhima dojke i krvnih sudova. Na osnovu funkcije raspodele ponderisanja, fluktuirajući fenomen se može ublažiti, smanjujući na taj način prekomerno poboljšanje korišćenjem gama korekcije.



Slika 3.5 *Ilustracija toka aktivnosti AGCWD metode na slici mamograije dojke. Sa desne strane slike su prikazani dijagrami distribuiranosti piksela pre i nakon obrade slike AGCWD metodom.*

4. GRAFIČKI-KORISNIČKI INTERFEJS

4.1.Specifikacija

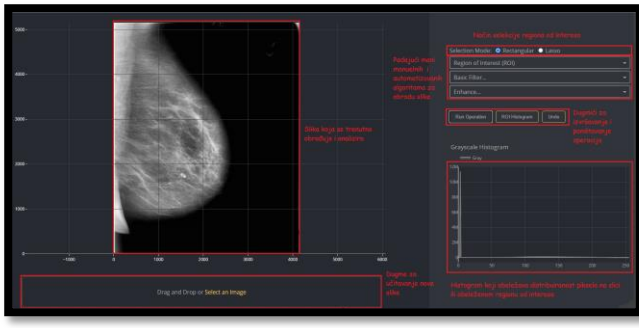
BreastAnalysisApp je aplikacija koja podržava kako manuelnu, tako i automatizovanu (izvan opsega ovog rada) obradu regiona od interesa i/ili čitave slike mamografije dojke.

Aplikacija je implementirana uz pomoć Python okvira, Dash, koji apstrahuje sve tehnologije i protokole neophode za izgradnju interaktivnih, analitičkih web aplikacija, slika 4.1.

4.2. Arhitektura

Arhitektura predloženog CAD (*engl. Computer Aided Diagnosis*) sistema se može grubo podeliti u dve celine:

1. **Manuelna manipulacija i analiza slike**. Pruža krajnjem korisniku mogućnost da na specifičan način manuelno obradi region od interesa, ili čitavu sliku.
2. **Automatizovana analiza mamografije dojke.** Pruža automatizovan mehanizam za otkrivanje abnormalnih pojava na mamografiji dojke, kao i mogućnost klasifikacije željenih regiona od interesa.



Slika 4.1. *Kratak pregled grafičkog korisničkog interfejsa i funkcionalnosti predloženog*

5. ZAKLJUČAK

Čitava svrha ovog istraživanja jeste da pruži motivaciju za daljim razvojem radioloških CAD (*engl. Computer Aided Diagnosis*) sistema baziranih, kako na tradicionalnim tehnikama računarske vizije, tako i na tehnikama mašinskog i/ili dubokog učenja, koje su detaljnije opisane u okviru master teze na osnovu koje je ovaj rad motivisan.

Danas, veliki broj studija se više fokusira na potpunu automatizaciju analize slike, nego na poboljšavanje postojećih, kao i razvoj novih CAD sistema koji će služiti, ne kao zamena, već kao pomoćno sredstvo u kliničkoj praksi. Čak i sa mnogim obećavajućim rezultatima ranih istraživačkih studija, postoji mnogo pitanja na koja treba dati odgovor pre uvođenja ovakvih sistema u radiološku praksu.

6. LITERATURA

[1] Rafael Gonzalez: Digital Image Processing
[2] Adaptive Histogram Equalization – Wikipedia
[3] Efficient Contrast Enhancement using Adaptive Gamma Correction and Cumulative Intensity Distribution
[4] Y. Kim, Contrast enhancement using brightness preserving bi-histogram equalization
[5] Y. Wan, Q. Chen, and B. Zhang, Image enhancement based on equal area dualistic sub-image histogram equalization method
[6] K. S. Sim, C. P. Tso, and Y. Tan, Recursive sub-image histogram equalization applied to gray-scale images
[7] M. Kim and M. G. Chung, Recursively separated and weighted histogram equalization for brightness preservation and contrast enhancement

Kratka biografija:

Stefan Radonjić rođen je 23.08.1995. godine u Beogradu. Godine 2010. završio je Osnovnu školu „Kosta Trifković“ u Novom Sadu. Srednju Ekonomsku Školu „Svetozar Miletić“ u Novom Sadu, završio je 2014. godine. Iste godine upisao je Fakultet tehničkih nauka na Univerzitetu u Novom Sadu, smer Računarstvo i Informatika. Zvanje diplomirani inženjer elektrotehnike i računarstva stekao je 2018. godine. Iste godine upisao je master akademske studije na smeru Računarstv i Automatika. Položio je sve ispite propisane planom i programom. Kontakt: stefan.radonjic995@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.3
DOI: https://doi.org/10.24867/15BE03Gotovac

IMPULSNI STRUJNI GENERATOR

PULSE CURRENT GENERATOR

Zdravko Gotovac, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu prikazana je jedna realizacija impulsnog strujnog generatora koji je namenjen za ispitivanje i odabiranje zavojnica čija je uloga prikupljanje elektromagnetne energije. Standardom su definisani talasni oblici strujnog impulsa kojima se vrši simulacija udara groma ili pražnjenja usled prenapona na dalekovodu. Generatori koji u potpunosti zadovoljavaju specifikacije su izuzetno skupi, njihova cena višestruko prevazilazi raspoloživi budžet. Zato je projektovan impulsni strujni generator čija je osnovna osobina ponovljivost talasnog oblika impulsa. Ovo se pokazalo kao dovoljno dobro rešenje u izboru optimalne zavojnice za prikupljanje električne energije, što i predstavlja sledeći korak u istraživanju.*

Ključne reči: *strujni impuls, RLC kolo, mikrokontroler*

Abstract – *This paper shows the realization of an pulse current generator whose purpose is to test and find an adequate coil which will be used for harvesting electric energy. Standard defines signal wave forms of current impulse which are used to simulate thunderbolt, or electrical discharge caused by overvoltage in high voltage transmission lines. Generators which completely fulfill required specifications are prohibitively expensive, and their price far exceeds available budget. This is the reason for designing pulse current generator whose main characteristic is repeatable pulse waveform. This has been shown as a sufficient solution in choosing optimal coil for harvesting electrical energy, which represents the next step in research.*

Keywords: *current pulse, RLC circuit, microcontroller*

1. UVOD

Struja koja protiče kroz provodnik stvara magnetno polje u njegovoj okolini. Ako se zavojnica postavi u promenljivo magnetno polje u njoj će, usled pojave električne indukcije, doći do pojave elektromagnetnog polja. Zavisno od oblika i dimenzija zavojnice, kao i od broja zavojaka, prikupljanje električne energije će biti manje ili više efikasno.

Struja kroz provodnik može biti jednosmerna ili naizmenična. Takođe, možemo posmatrati ustaljeni ili prelazni režim rada. Trenutno se u okviru istraživanja raspolaže skupom zavojnica koje treba ispitati i ustanoviti koja je najefikasnija u prikupljanju električne energije.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Dragan Pejić.

U prvoj fazi su zavojnice ispitivane u ustaljenom naizmeničnom režimu. Obavljena su ispitivanja pri struji efektivne vrednosti od 30 A i frekvenciji 3 kHz. Neki od razloga zbog kojih je ispitivanje u ustaljenom režimu nedovoljno dobro su:

a) na odziv zavojnice utiče efektivna vrednost struje, ali i vrednost prvog izvoda struje, što je direktno srazmerno frekvenciji

b) zavojnica će biti korišćena u praksi u impulsnom, a ne u ustaljenom režimu.

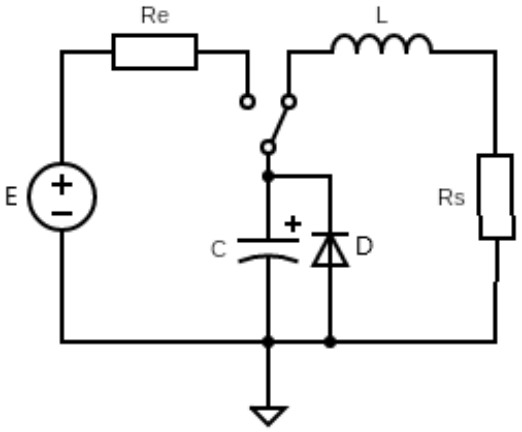
Kako bi se obezbedilo korektnije ispitivanje zavojnica, projektovan je impulsni strujni generator. Cilj projektovanja je napraviti strujni impuls veće vrednosti struje (od raspoloživih 30 A), sa brzinom porasta i opadanja ivice većom nego pri frekvenciji od raspoloživih 3 kHz i sa ponovljivim talasnim oblikom.

Olakšavajuća okolnost je što impulsni strujni generator ne treba da proizvede i veliki napon (a time i veliku snagu), što ga razlikuje od standardnih rešenja koja postoje, ali su izuzetno skupa: zavisno od snage, njihova cena se kreće od nekoliko hiljada do nekoliko stotina hiljada evra.

2. SIMULACIJA, IZRADA I ISPITIVANJE KOLA

2.1. Teorijski opis funkcionisanja kola

Da bi se shvatio rad čitavog kola, prvo je potrebno upoznati se sa osnovnom šemom koja objašnjava bazično kolo (slika 1), na kom se zasniva rad ispitanog kola, odnosno na pražnjenju kondenzatora kroz zavojnicu i otpornik (RLC kolo).

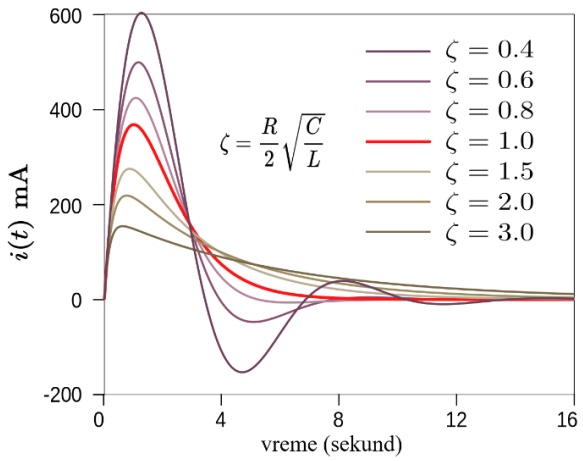


Slika 1. *Osnovna šema kola*

Na slici 1. možemo da vidimo osnovne komponente kao što su izvor napajanja E, otpornici Re i Rs, zavojnica L, kondenzator *C*, i zaštitna dioda D.

Kondenzator se prvo napuni kroz otpornik Re iz jednosmernog izvora (preklopnik u levom položaju). Onda se preklopnik prebaci u desni položaj, pa se kondenzator prazni kroz zavojnicu L i otpornik Rs.

Uzimajuću u obzir ograničenja napajanja koje je povezano na kondenzatore, kada je preklopnik u levom položaju (slika 1), struja kojom se kondenzator puni nije velika, te se sporo puni. Kako je ekvivalentna otpornost desnog dela kola mala, kada je preklopnik u desnom položaju (slika 1), struja je velika. Dakle to su dva razloga zbog kojih se kondenzatori u kolu sporo pune, i brzo prazne (u vidu impulsa).



Slika 2. *Strujni odzivi kola, kao funkcija upotrebljenih komponenti*

Kako je potrebno uobličiti impulsni signal, odnosno uzlaznu ivicu strujnog signala, u kolo je ubačena zavojnica L, koja sprečava brze promene u jačini struje unutar kola.

Odzivi koji se mogu videti na slici su posledica faktora prigušenja (ζ). Kada je faktor prigušenja manji od jedan (ζ<1), onda je reč o slabo prigušenom kolu. Za kolo čiji je faktor prigušenja veći od jedan (ζ>1), kažemo da su dobro prigušena. Postoji i poseban slučaj, a to je kada je faktor prigušenja jedan jedinici (ζ=1), i tada je reč o kritično prigušenom kolu. Faktor prigušenja zavisi od vrednosti komponenti RLC kola.

Strujni odzivi slabo prigušenog kola su prikazani na slici 2. kao krive ljubičastih boja, dok su strujni odzivi dobro prigušenog kola predstavljeni u vidu krivih oker boje. Za ispitano kolo idealan bi bio strujni odziv na slici prikazan crvenom bojom, odnosno odziv kritično prigušenog kola, pa je kolo napravljeno tako da ima taj odziv.

Zbog nesavršenosti električnih elemenata korištenih za pravljenje kola, može se desiti da kolo bude slabo prigušeno, što može dovesti do proticanja struje u suprotnom smeru, što bi oštetilo kodnezatore.

Da bi se sprečilo potencijalno oštećenje kondenzatora dodata je dioda D.

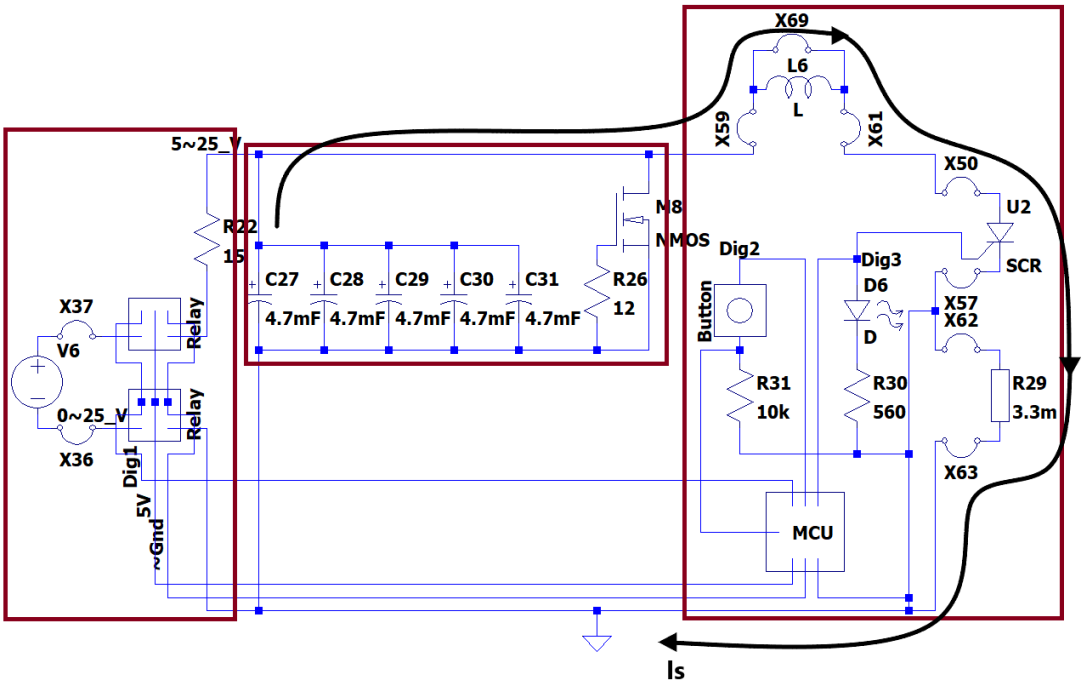
2.2. Simulacija i skiciranje kola

Konačna šema čitavog kola može se videti na slici 3, na kojoj se mogu uočiti obeležena tri glavna dela šeme. Prvi deo šeme (gledano sa leva na desno) predstavlja deo kojim se kontroliše napajanje čitavog kola. Mogu se videti releji obeleženi sa Relay, džamperi X36 i X37, otpornik R22 i naponski izvor V6.

Džamperi predstavljaju terminale na konektoru na koje se povezuje napajanje V6, i to je spoljno dovedeno napajanje. Kolo se snabdeva jednosmernim naponom koji se može menjati od 0 V do 30 V. Prilikom testiranja kolo nije napajano naponom većim od 25 V, i to je bilo dovoljno za ispitivanje osobina kola. Napajanje je redno povezano sa sledećim delom kola, otpornikom *R22*.

Sledeći deo kola predstavlja pet kondenzatora koji su povezani u paralelnu vezu. Kako je svaki od kondenzatora iste nazivne kapacitivnosti, ukupna ekvivalentna kapacitivnost dobija se formulom:

$$Cu = C27 + C28 + C29 + C30 + C31 = 5 \times C27 \quad (1)$$



Slika 3. *Šema kompletnog kola*

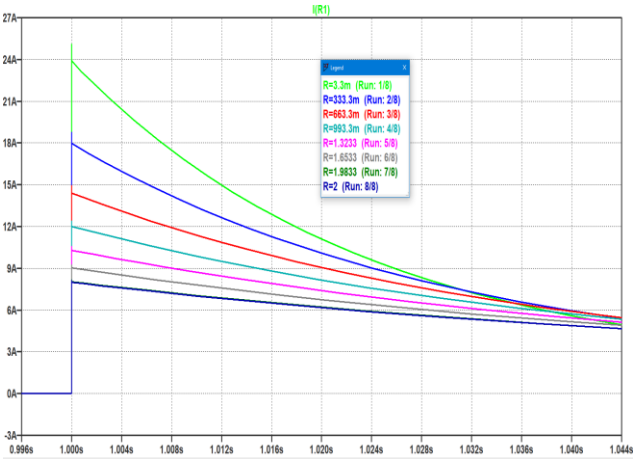
Paralelno sa kondenzatorima nalazi se povezan MOSFET M8, koji je n-kanalnog tipa (NMOS). Korišteni MOSFET je 26NM60N, koji može da izdrži konstantni napon (VDS) između drejna (engl. Drain) i sorsa (engl. Source) u visini od 600 V, i konstantnu struju drejna (Id) u visini od 20 A. Otpornik R26 drži NFET u isključenom stanju, tako da do izraza dolazi samo ugrađena zaštitna antiparalelna dioda, koja odgovara diodi sa slike 1.

Na pozitivni kraj diode povezan je terminal, na šemi predstavljen džamperima X59 i X61, i on je prva komponenta sledećeg, i poslednjeg dela kola. Terminal je postavljen tako da se na njegove konektore može povezati kalem L6, ili kratkospojnik X69, u zavisnosti od toga koja se konfiguracija komponentni unutar kola želi ispitati. Na kalem se redno povezuje tiristor U2, koji ima ulogu prekidačke komponente.

Na šemi su dodati džamperi da se naglasi gde se sve vrše spajanja spoljašnjih komponenti na štampanu ploču. Svako takvo spajanje je potencijalno povećanje ukupne otpornosti u kolu, a to može bitno da promeni (smanji) maksimalnu vrednost struje.

Mikrokontroler (MCU) upravlja prekidačkim komponentama (tiristorom i relejima), tasterom Button kojim se pokreće proces električnog pražnjenja, koja je kao program smeštena na mikrokontroler, i signalna LED koja se koristi kao indikator izvršavanja određenog dela programa.

Kada se kolo poveže na spoljašnje napajanje, i kada se mikrokontroleru obezbedi napajanje, pritiskom na taster, i detekcijom visokog nivoa na ulaznom pinu mikrokontrolera, mikrokontroler počinje proces električnog pražnjenja koji se nalazi na njemu.



Slika 4. *Veza odziva impulsnog signal i otpornosti kola*

Prvo se uključuju releji, koji u tom slučaju spajaju napajanje sa ostatkom kola, na način koji je pokazan na šemi. Taj deo programa se izvršava određeno vreme, koje može biti izmenjeno u zavisnosti od potrebe. To vreme izvršavanja dela programa zavisi od vremena potrebnog da se napune kondenzatori. Vreme punjenja kondenzatora zavisi od otpornika kroz koji protiče struja od izvora ka pozitivnim pinovima kondenzatora, kao i od kapacitivnosti samih kondenzatora, budući da proizvod te dve veličine definiše vremensku konstantu:

$$\tau = RC \tag{2}$$

Kada se kondenzatori napune releji se isključuju, i tako su kondenzatori u potpunosti galvanski izolovani od izvora napajanja, i ovo je neophodno da se uradi, jer se prilikom testiranja primetio veliki šum na kanalu jedan (grafik 1, CH1), koji je sprečavao ispravnu akviziciju signala. Napunjeni kondenzatori se koriste kao baterija u sledećem delu izvršavanja programa.

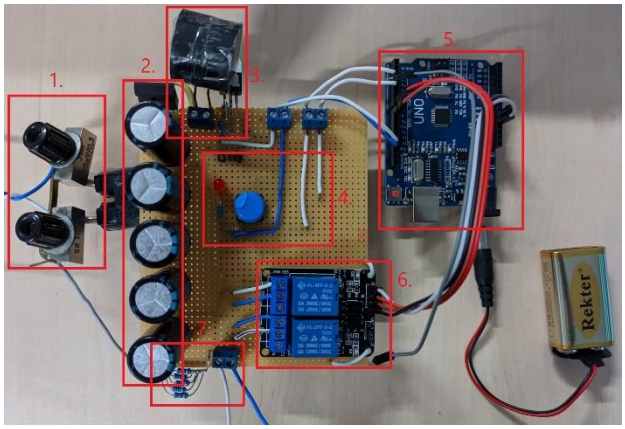
Sledeći korak je paljenje tiristora, koji se nakon paljenja ponaša kao kratak spoj, i ostvaruje vezu između jednog kraja terminala na koji je povezana zavojnica i šanta (otpornik R29).

Bitno je napomenuti da je vrednosti otpornosti šanta reda mΩ, dok je otpornost vodova i kontakata značajno veća, reda Ω. Taj problem se izbegava četvorožičnim povezivanjem. Šant se povezuje četvorožično pa je lako izračunati struju koja prolazi kroz njega.

Prilikom ostvarivanja veze između terminala na kojima se nalazi zavojnica i šanta, kondenzatori se brzo isprazne, u vidu strujnog impulsa. Struja nastala prilikom pražnjenja kondenzatora može se videti na slici 4.

2.3. Izrada kola

Realizovano kolo može da se vidi na slici 5. Za bazu kola korištena ja bakarna pločica na kojoj se nalaze zalemljenje komponente. Može se videti terminal na koji je povezan šant (1), deo kola na kom su paralelno povezani kondenzatori i MOSFET (2), terminal na koji je povezana zavojnica i na njega redno tiristor (3), kolo sa tasterom i signalnom LED (4) i terminal na koji se povezuje napajanje, i na njega redno povezana grupa paralelnih otpornika (7).



Slika 5. *Realizovano električno kolo*

Na slici se može videti da je kao mikrokontroler korištena Arduino UNO razvojna platforma (5), i pločica na kojoj se nalaze dva zalemljena relejna kola (6).

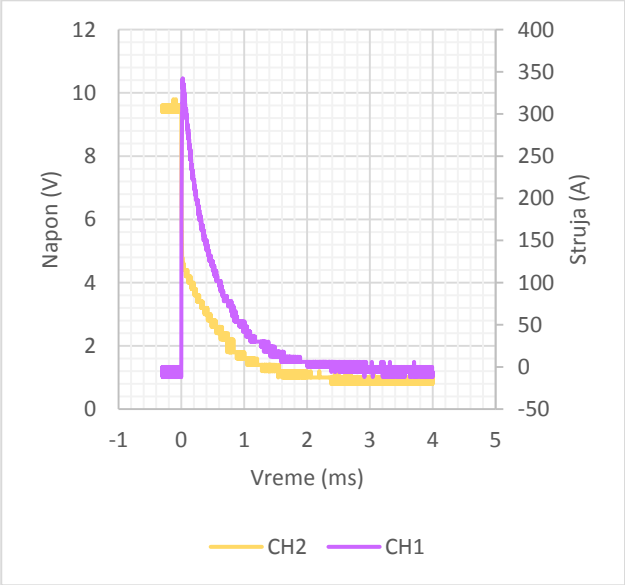
Napajanje Arduino UNO razvojne platforme, kao i kola sa tasterom i signalne LED obezbeđeno je baterijom od 9 V.

2.4. Ispitivanje signala unutar kola

Mereni signali nalaze se prikazani na grafiku 1, gde se pored signala koji je meren na kanalu jedan (CH1), a koji predstavlja signal na šantu, vidi i signal meren na kanalu 2 (CH2), koji predstavlja struju koja prolazi kroz šant.

Početak snimanja signala je obavljen u trenutku nakon punjenja kondenzatora, i kratak vremenski period nakon njihovog pražnjenja. Može se videti kako se napon kondezatora na početku naglo menja, jer je u tom trenutku najveća struja koja prolazi kroz šant. Sa smanjivanjem

napona na pozitivnom kraju kondezatora (negativni kraj je povezan na GND), smanjuje se i struja koja prolazi kroz šant.



Grafik 1. *Signalni odzivi unutar kola, napon na kondenzatorima (CH1) i struja kroz otpornik Rs (R26)*

Tako dobijamo impulsni signal koji ima karakterističan talasni oblik koji zavisi od komponenti od kojih se sastoji kolo (kondezatora, kalema i otpornika). Signal na krajevima šanta brzo dostiže svoj maksimum, i na grafiku se može videti, da kada struja unutar kola dostigne svoj maksimum, iznosi ~342 A.

Kolo dostiže ovaj maksimum za manje od 100 µs, i kao i na jačinu struje, na brzinu odziva signala može se uticati različitim konfiguracijama elemenata unutar kola.

3. ZAKLJUČAK

U radu je dat prikaz realizacije jednostavnog impulsnog generatora.Prikazani su rezultati teorijske i simulacione analize, ali i rezultati merenja na relizovanom prototipu.

Uprkos rezultata koji su dobijeni teorijskom obradom, i simulacijama, koje su pokazale, impulsni generator je napravljen od realnih komponenti, pa je kao takav posedovao određene nesavršenosti.

Otpornost provodničkih veza između komponenti su predstavljale značajnu razliku u odnosu na ono što su simulacije pokazivale. Zbog njihovih otpornosti, koje su bile značajno veće, u poređenju sa otporničkim elementom (šant) koji se nalazi unutar realizovanog kola, strujni impuls koji se javljao tokom pražnjenja je bio značajno slabiji.

Unutrašnje kapacitivne, kao i induktivne, osobine komponenti uticale su na talasni oblik impulsnog signala koji se generisao prilikom pražnjenja kondenzatora.

Ali kao što je pre početka rada predpostavljeno, u radu se pokazala realnom mogućnost realizacije impulsnog strujnog generatora.

Na veličinu strujnog impulsa se može uticati povećanjem napona napajanja i smanjenjem otpornosti u kolu. Trenutno korišteni kondenzatori su predviđeni za maksimalni napon od 35 V. Otpornost u kolu se može smanjiti kvalitetnijom izradom vodova kroz koje protiče velika struja. Ukoliko se želi postići brža uzlazna ivica, potrebno je smanjivati vrednost induktivnosti.

Sa jednostavnim komponentama, napravljen je impulsni strujni generator malih dimenzija, koji je u stanju da generiše ponovljivi signal, a da je taj strujni impulsni signal ima izuzetno kratak vremenski odziv (reda par milisekundi), i veliku jačinu struje (reda nekoliko stotina ampera).

Kratka biografija:

Zdravko Gotovac rođen je u Novom Sadu 1997. god. Bachelor rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Merenje i regulacija odbranio je 2020.god.
kontakt: gotovaczdravko17@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.39
DOI: https://doi.org/10.24867/15BE04Bilinac

PREGLED 4G LTE SISTEMA I TEHNIKE LOKALIZACIJE

4G LTE SYSTEM AND LOCALIZATION TECHNIQUES

Nemanja Bilinac, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *4G LTE sistem napravio je velike promjene u ICT tehnologiji. Prethodni mrežne generacije nisu ni blizu 4G LTE sistemu u pogledu zadovoljenja potreba krajnjih korisnika. Samim tim 4G LTE je postavio visoke standarde u pogledu daljeg razvoja ove oblasti, kao i sve veće zahtjeve krajnjih korisnika. Tu bitnu ulogu imaju razne tehnike lokalizacije u okviru 4G LTE sistema.Na taj način jasno se nameće činjenica da je 4G LTE stadnard jedna od ključnih stavki u pogledu nove industrijske revolucije, koju će činiti novonastupajući mrežni standardi u vidu 5G sistema, koji još nije ušao u potpunu komercijalnu upotrebu na globalnom nivou.*

Ključne reči: *4G LTE, Tehnike lokalizacije*

Abstract – *The 4G LTE system brought major changes in ICT technology. Previous network generations are nowhere near 4G LTE systems in terms of meeting the needs of end users. Therefore 4G LTE has set high standards in terms of further development of these areas, as well as the growing demands of end users. Different localization techniques within 4G LTE system play important role. Due to this, it is clearly stated that the 4G LTE standard is one of the key items in terms of new industrial revolutions, which will be made of the emerging network standards in the form of a 5G system that has not yet entered full commercial use globally.*

Keywords: *4G LTE, Localization techniques*

1. UVOD

Evidentna je sve veća težnja ka komunikaciji mašina-mašina ili čovjek-mašina, sve većoj upotrebi uređaja poput dronova koji imaju funkciju pokretnih baznih stanica, senzora koji prikupljaju neke podatke iz okoline u kojoj se nalaze koji takođe mogu biti mobilni u okviru neke bežične mreže i još mnogo drugih lokaciono baznih orijentisanih servisa. Prava ekspanzija se tek očekuje što nesumnjivo pozicionira komunikacione tehnologije, kao i lokalizaciju među ključne faktore za pravilno izvršenje funkcija koje moderni servisi zahtjevaju.

Upravo je lokalizacija sa fokusom na 4G LTE sistem glavna tema ovoga rada. U radu će pored opštih tehnika koje se koriste u lokalizaciji mrežnih čvorova biti riječi o infrastrukturi koja podržava ove lokalizacione tehnike, kao i smetnje koje svakako otežavaju pravilnu procjenu pozicije čvorova u okolini. Na kraju rada prikazan je eksperiment pomoću RSS lokalizacione tehnike.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dejan Vukobratović, red. prof.

2. LOKALIZACIONE TEHNIKE

Lokalizacione tehnologije možemo da podjelimo u dvije glavne kategorije, a to su:

- Lokalizaciona tehnologija bazirana na satelitu
- Lokalizaciona tehnologija bazirana na mrežnoj infrastrukturi [1].

Lokalizacione tehnologije bazirane na satelitu se uglavnom koriste u slučaju lokalizacije na otvorenom prostoru. Najrasprostranjeniji ovakav sistem koji se najviše koristi je GPS (eng, *Global Positioning System*) lokalizacioni sistem (Slika 1).

Slika 1. *Infrastruktura GPS Sistema*

Kod lokalizacionih tehnologija koje se isključivo baziraju na mrežnu strukturu prvenstveno je potrebno odrediti rastojanje između čvorova mreže.

Za određivanje rastojanja između čvorova mreže, postoje razne metode, a to su:

- Lokalizacija na osnovu vremena prijema (Time of arrival –ToA)
- Lokalizacija na osnovu vremenske razlike prijema (Time Difference of Arrival – TDoA)
- Lokalizacija na osnovu mjerenja dolaznog pravca (Angle of arrival – AoA)
- Lokalizacija na osnovu snage prijemnog radio signala (Received Signal Strength – RSS) [2].

TDoA tehnika lokalizacije je opisana posebno kao zasebno poglavlje, jer je to jedna od najzastupljenijih tehnika lokalizacije u praksi.

3. LOKALIZACIONA INFRASTRUKTURA

Validnost procjenjene lokacije nekog mobilnog čvora u mreži u velikoj mjeri zavisi od tipa infrastrukture koja omogućava lokalizaciju.

Ističu se dva načina implementacije u pogledu lokalizacione infrastrukture, a to su:

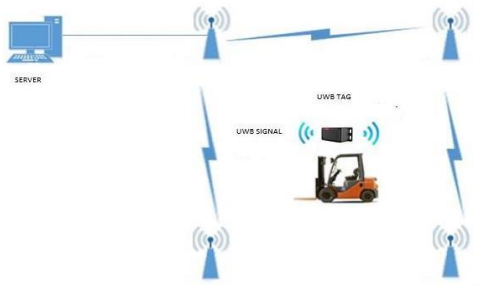
- Izgradnja posebno namjenjene struktrure za lokalizaciju
- Iskorišćenje već postojeće infrastrukture čija je prvobitna namjena komunikacioni servis.

Kada je riječ o prvom načinu implementacije glavna pozitivna stvar je da se mogu ostvariti visoke lokalizacione perfomanse koje se mogu ostvariti korišćenjem specijalnog referentnog signala, kao i specijalizovano namjenjeni hardver za to. Što se tiče negativnih strana tu se ističu veoma skup hardver i ograničenja u sistemskoj skalabilnosti. Što se tiče drugog načina, pozitivna strana je ta da se nema skupih hardverskih komponenti koje pored toga zahtevaju i dosta vremena za njihovo pravilno raspoređivanje. Negativna strana je ta da se zahtjevaju dodatni sofisticirani algorimi, da bi se poboljšale perforamnse samoga sistema. U tom slučaju kompleknsot algoritma i njegova kohezija sa već postojećom infrastrukturom predstavljaju velike izazove.

Lokalizacione struktrure koje se najčešće koriste su:
- GNSS (eng. Global Navigation Satellite System)
- Ćelijska mrežna infrsatruktura
- Wi-Fi infrsatruktura
- Lokalizacioni sistem baziran na UWB (eng. Ultra Wideband) tehnologiji

Kada je u pitanju pokrivenost područja, svaka od navedenih infratsruktura ima svoje parametre. U okviru GNSS sistema pokrivnsot je globalna, ali je preciznost veoma diskutabilna u nekim gusto naseljenim sredinama i samim tim sporna za neke osjetljive aplikacije. Zbog toga se koriste druge lokalizacione strukture. Kod ćelijske strukture pokrivenost po jednom fiksnom čvoru varira od 25 do 500 metara u zavisnosti koja je generacija u pitanju. Naravno kako je gušći spektar, sve veći zahtjev korisnika i pokrivenost opada, odnosno ima sve više fiksnih čvorova. Wi Fi infrastruktura (Slika 2) kao i UWB tehnologija (Slika 3) koriste se za neka lokalna okruženja.

Slika 2. Wi-Fi infrastruktura



Slika 3. UWB tehnologija

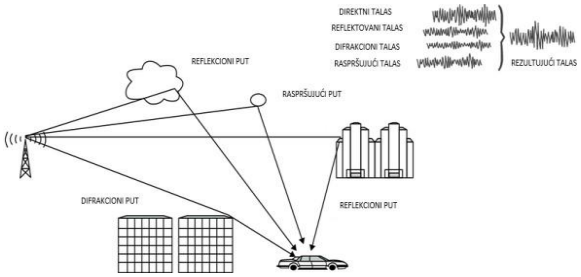
4. GLAVNI IZVORI GREŠAKA

Jasno je da nije moguće izvršiti stoprocentno tačnu procjenu lokacije, ali treba nastojati da negativan uticaj izvora grešaka, kao i neizbježan uticaj pristrasnosti prilikom mjerenja lokacije mobilnog čvora bude sveden na što je manje mogući nivo. Danas je ovo posebno veliki izazov imajući u vidu činjenicu da sve veći broj aplikacija zahtjeva što bržu i precizniju procjenu lokacije, odnoso real-time praćenje uređaja.

Tri glavna izvora grešaka su:

- Feding
- Sistemske greške
- Propagacije u uslovima bez optičke vidljivosti.

Feding je jedan od izvora grešaka koji su posebno karakteristični bežičnom kanalu i predstavlja izuzetno veliki nivo degradacije signala prilikom njegovog prostiranja od predajnika do prijemnika. Sredina u kojoj je to posebno prisutno je gradska zona u kojoj ima dosta prepreka u vidu drveća, zgrada i mnogih drugih. (Slika 4)



Slika 4. Višestruki feding

Problem NLoS je najviše prisutan u gusto naseljenim gradskim zonama, gdje se nalazi veliki broj prepreka koje izazivaju razne dodatne komponenete originalnog signala i dobijanje superponiranog signala na prijemu što svakako nije poželjno. Pojave koje su posljedica uslova bez optičke vidljivosti su refleksija, refrakcija, difrakcija i rasijanje.

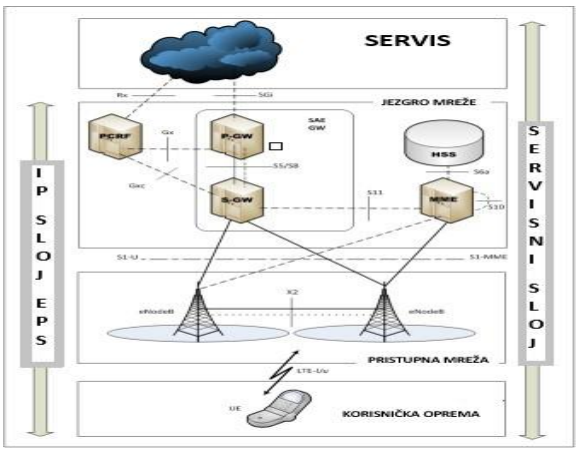
Sistemski izvor grešaka se jako često dešava u relanim sistemima za procjenu lokacije. Ovaj tip grešaka je generisan unutar samoga sistema, a ne od nekog spoljašnjeg faktora kao što je bio slučaj sa prethodna dva izvora grešaka [3].

5. LONG TERM EVOLUTION (LTE)

Četvrta generacija mobilne komunikacione mreže odnosno LTE, predstavljala je veliki tehnološki napredak u pogledu mobilnih komunikacija. Razvoj 4 generacije se može podijeliti na **standardni LTE** i **napredni LTE.**

Kada je u pitanju standardni LTE, on je bio smatran standardom u okviru 3GPP (*eng. 3rd Generation Partnership Project)*, koji je bio pripremna faza u okviru treće generacije mobilne komunkacione mreže za četvrtu generaciju. Proces intenziviranog razvoja LTE standard je počeo 2008 godine, sa posebnim fokusom na IP jezgro mreže, OFDMA (*eng. Orthogonal Frequency Division Multiple Access)* interfejsu, poboljšanju HSDPA (*eng. High Speed Downlink Point Access)* i HSUPA (*eng. High Speed Uplink Point Access)* protokola kao i unaprijeđenje MSR (eng. Multi Standard Radio) segmenta.

Time je dobijena dosta kompaktnija i poboljšanija sveukupna mrežna arhitektura standardnog LTE (Slika 5).



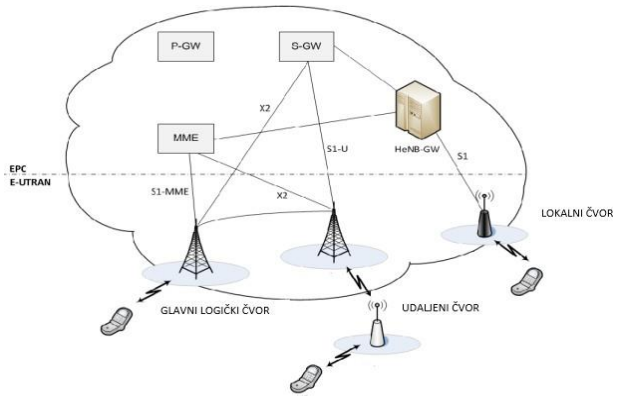
Slika 5. Arhitektura standardnog LTE

Kao rezultat stalnog rasta u količini podataka, povećanje želje krajnjih korisnika za novim servisima krenulo se u težnju za unaprijeđenjem radio interfejsa.

Praktično je ova faza razvoja krenula da se realizuje od 2011 godine, kada je i zvanično puštena u komercijalnu upotrebu četvrta generacija mobilnih komunikacija. LTE standard u okviru ove generacije je prešao sa standradnog nivoa na novi, poboljšani napredni LTE.
Kao glavni ciljevi naprednog LTE bili su:

- Fleksibilno i brže postavljanje mreže
- Bolja pokrivenost i unaprijeđena spektralna efikasnost
- Veća fleksibilnost u pogledu raspodjela širokopojasnog spektra
- Sveprisutnost i isplativost širokopojasne mreže [4]

Ovim je dobijena još kompaktnija mrežna arhitektura naprednog LTE standarda (Slika 6).



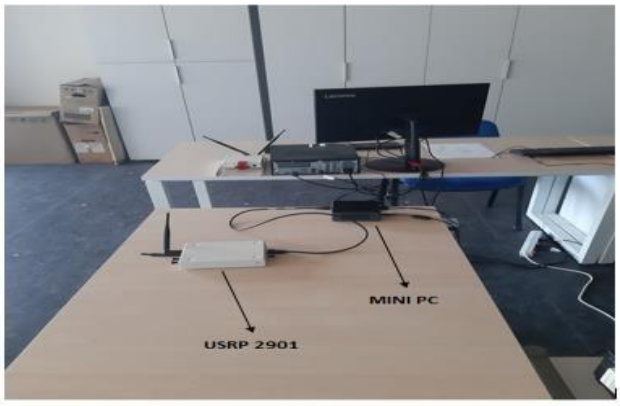
Slika 6. Arhitektura naprednog LTE

6. TDoA (Time Difference of Arrival)

Najveća razlika u odnosu na ToA lokalizacionu tehniku jeste činjenica da se ne posmatra distancu između svakog fiksnog čvora i prijemnika, već se to posmatra sa aspekta para fiksnih čvorova u odnosu na prijemnik.Osiguranje kvaliteta servisa u 4G LTE mreži, predstavlja veliki izazov za sve one koji su uključeni u proces planiranja, implementiranja i održavanja mreže.
Uz pomoć TDoA tehnike se u okviru 4G LTE mreže, može izvršiti lokacija korisnika na udaljenosti manjim od 100 metara sa dosta velikom pouzdanošću.
Bitno je istaći da i pored toga što je TDoA među najčešće korišćenim lokalizacionim tehnikama, u praksi se ona veoma često kombinuje sa nekim drugim lokalizacionim tehnikama [5].

7. EKSPERIMENTALNI DIO

U okviru ovoga poglavlja detaljno je opisan praktični dio rada. Fokus je na unutrašnjoj pokrivenosti prostorije signalom, korišćenjem OpenAirInterface softverske platforme kao i Nokia EPC Core Network softvera pomoću kojih se podiže mini bazna stanica. OpenAirInterface je kao softverksa platforma podignut na PC-u, dok je Nokia EPC Core Network softver kao podrška pored 4G LTE mreže i za novonastupajuću 5G mrežu, podignut na mini PC-u. Sve to je spušteno na jedan USRP (eng.Universal Software Defined Radio) uređaj model 2901, koji emituje signal unutar prostorije i simulira jednu realnu 4G LTE baznu stanicu.
Mjerenja sa unaprijed utvrđenih pozicija u prostoriji su izvršena korišćenjem mobilnog telefona putem Electro Smart aplikacije, koji je uspostavio konekciju sa mini baznom stanicom (Slika 8).
Na slici 7 prikazana je oprema koja je korišćena u izvođenju eksperimentalnog dijela rada.



Slika 7. Eksperimentalna oprema

Slika 8. Mjerne pozicije

Sama aplikacija daje vrijednosti signala preko RSSI vrijednosti izraženim u dbm. Na ovaj način se preko RSS tehnologije, simulira rad jedne stvarne 4G LTE stanice. U okviru OpenAirInterface softvera, preko glavnog PC smo podesili parametre vezane za MCC (eng. Mobile Country Code) i MNC (eng. Mobile Network Code). Pošto svaka država ima svoj rezervisani MCC i MNC za svaki svoj operator, vrijednosti koje su stavljene za MCC

i MNC su 901, 70 respektivno i nisu parmetri niti jedne države na svijetu jer su to rezervisane vrijednosti. Naša bazna stanica predstavlja mini mrežu pod nazivom ICONIC, i koristi se za neke lokalne eksperimentalne svrhe.
IP adresa koja je dodijeljena linku za komunikaciju, odnosno baznoj stanici je 192.184.0.1, dok je adresa dodjeljena mobilnom telefonu 192.184.0.1. Sve ovo je funkcija Nokia EPC Network Core softvera. Proces komunikacije prikazan je na slici 9.

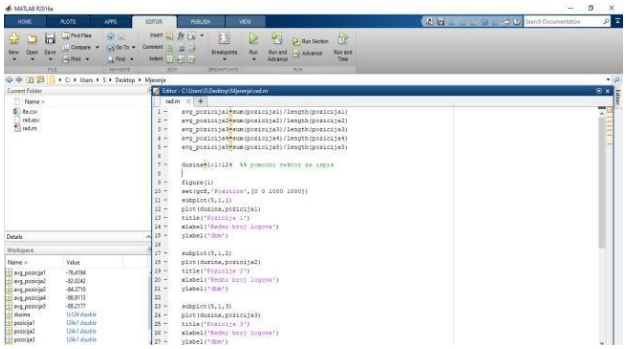
```
192.184.0.2 ICMP          100 Echo (ping) request  id=0x15e0, seq=487/59137, ttl=64 (reply in 662)
192.184.0.2 GTP <ICMP>    136 Echo (ping) request  id=0x15e0, seq=487/59137, ttl=64 (reply in 661)
192.184.0.1 GTP <ICMP>    136 Echo (ping) reply    id=0x15e0, seq=487/59137, ttl=255 (request in 660)
192.184.0.1 ICMP          100 Echo (ping) reply    id=0x15e0, seq=487/59137, ttl=255 (request in 659)
192.184.0.2 ICMP          100 Echo (ping) request  id=0x15e0, seq=488/59393, ttl=64 (reply in 666)
192.184.0.2 GTP <ICMP>    136 Echo (ping) request  id=0x15e0, seq=488/59393, ttl=64 (reply in 665)
192.184.0.1 GTP <ICMP>    136 Echo (ping) reply    id=0x15e0, seq=488/59393, ttl=255 (request in 664)
192.184.0.1 ICMP          100 Echo (ping) reply    id=0x15e0, seq=488/59393, ttl=255 (request in 663)
192.184.0.2 ICMP          100 Echo (ping) request  id=0x15e0, seq=489/59649, ttl=64 (reply in 670)
192.184.0.2 GTP <ICMP>    136 Echo (ping) request  id=0x15e0, seq=489/59649, ttl=64 (reply in 669)
192.184.0.1 GTP <ICMP>    136 Echo (ping) reply    id=0x15e0, seq=489/59649, ttl=255 (request in 668)
192.184.0.1 ICMP          100 Echo (ping) reply    id=0x15e0, seq=489/59649, ttl=255 (request in 667)
```

Slika 9. Komunikacija između mini bazne stanice i mobilnog telefona

Na slici 10 prikazani se logovi koje je uhvatila aplikacija tokom mjerenja.

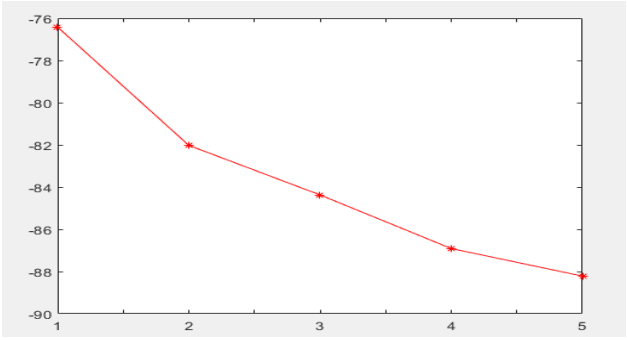
Slika 10. Logovi uhvaćeni od strane aplikacije

Obrada prikupljenih podataka urađena je u Matlab softverskom alatu. (Slika 11)



Slika 11. Rad u Matlabu

Posebna pažnja je posvećena dobijanju srednje vrijednosti signala sa svake mjerne pozicije u prostoriji (Slika 12).



Slika 12. *Srednje vrijednosti signala sa različitih mjernih pozicija*

8. ZAKLJUČAK

Novi trendovi u svijetu svakako će ovu oblast oblikovati tako da čini izuzetno bitan faktor u implementaciji novih tehnoloških dostignuća.
Kada je u pitanju eksperimentalni dio rada jasno je da i u zatvorenom prostoru postoji slabljenje signala udaljavanjem prijemnika od predajnika. Bitna je napomenuti činjenicu da je prostorija u kojoj je vršeno mjerenje dužine oko 15-ak metara.
Razdaljina koju smo obuhvatili izabranim mjernim pozicijama je oko 6 do 7 metara, što nam jasno govori da je pola prosorije uzeto u analizu pokrivenosti signala koje emituje naša bazna stanica. Dakle postoji slabljenje signala ali je taj nivo opadanja dosta manji od spoljašnjeg scenarija.

9. LITERATURA

[1] Drawil, Nabil M., Haitham M. Amar, and Otman A. Basir. "GPS localization accuracy classification: a context based approach."
[2] O. Bialer, D. Raphaeli, and A. J. Weiss, "Maximum- likelihood direct position estimation in dense multipath"
[3] C.Gentner, T.Jost, W.Wang, S. Zhang, A. Dammann, and U. –C. Fiebig," Multipath assisted positioning with simultaneous localization and mapping"
[4]] Akyildiz I. F., "*The Evolution to 4G cellular systems: LTE-Advanced"*
[5] Roberts, R., "*Tdoa localization techniques*"

Kratka biografija:

Nemanja Bilinac rođen je u Srbinju 1996. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Komunikacione tehnologije i obrada signala odbranio je 2021.god. kontakt: nemanja.bilinac96@gmail.com

Зборник радова Факултета техничких наука, Нови Сад

UDK: 004.8
DOI: https://doi.org/10.24867/15BE05Beker

ЗАМЕНЕ ЗА TCP ПРОТОКОЛ

ALTERNATIVES FOR TCP PROTOCOL

Александар Бекер, *Факултет техничких наука, Нови Сад*

Област – ПРИМЕЊЕНЕ РАЧУНАРСКЕ НАУКЕ И ИНФОРМАТИКА

Кратак садржај – *Обрада особина TCP протокола, кратак осврт на алтернативе истог и начине на који они превазилазе недостатке TCP-а. Акценат је стављен на QUIC протокол, где су описане особине протокола, док је начин функционисања приказан кроз библиотеку LSQUIC и клијентску апликацију која је развијене за потребе задатка.*

Кључне речи: *TCP, UDP, QUIC, protokol, DCCP, SCTP, MPTCP, lsquic*

Abstract – *Process the features of the TCP protocol, look at alternatives to it and the ways in which they overcome the disadvantages of TCP. Emphasis will be placed on the QUIC protocol, where the properties of the protocol will be described, and the way it works will be shown through the LSQUIC library and the client application which is developed for the needs of the task.*

Keywords: *TCP, UDP, QUIC, protocol, DCCP, SCTP, MPTCP, lsquic*

1. УВОД

Задатак овог рада јесте да се обради протокол четвртог нивоа *OSI (Open Systems Interconnection)* модела и то конкретно *TCP* протокол, обраде недостаци и размотре друга могућа решења. Развијен 1974. године, а данас доминантан транспортни протокол, свакако да има својих добрих страна, али исто тако има и мана које ће бити побројане и анализиране. Временом су настали и други протоколи који превазилазе одређене недостатке *TCP* (*Transmission Control Protocol*) протокола, а енкапсулирају исте или сличне функционалности. Неки од тих протокола биће наведени и обрађени у овом раду.

У првом поглављу дат је кратак увод у рад. У другом поглављу је детаљније описан начин на који ради *TCP*, његове добре стране, као и мане овог протокола. Треће поглавље односи се на опис протокола четвртог нивоа *OSI* модела, који су одабрани као подобне замене, где су такође описане добре стране тих протокола и које мане *TCP* протокола надомешћују. У четвртом поглављу, описан је *QUIC* протокол, који је производ компаније *Google* и који је званично одобрен маја текуће године.

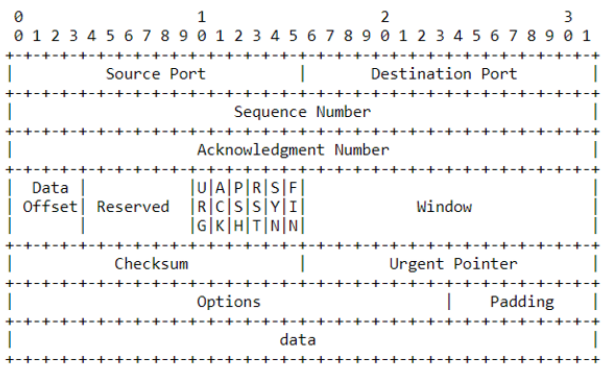
Након тога, у петом поглављу дат је закључак, којим смо сумирали претходно урађено и предложили даље кораке развоја рада.

2. *TCP* ПРОТОКОЛ

TCP протокол је транспортни протокол који омогућава комуникацију између два удаљена процеса, која су идентификована портовима. Предуслов за транспорт података је отварање конекције између две комуникационе стране.

TCP протокол групише одређене бајтове података у сегмент, а затим сегмент користи као основна јединицу преноса података. Сегменти се енкапсулирају у *IP* пакет, а затим се транспортују кроз мрежу.

Сваки сегмент садржи део са подацима и обавезан део који се назива заглавље сегмента. Дешава се и да сегменти не садрже податке, уколико се користе за потврђивање пријема других сегмената. Заглавље сегмента садржи обавезна и опциона поља, тако да сама дужина није фиксна. На слици 1 приказан је формат заглавља *TCP* сегмента



Слика 1. *Формат заглавља TCP сегмента [1]*

Пре слања података неопходно је да се успостави комуникациона веза, процесом који се назива *three-way handshaking*. Обе стране, пријемна и предајна, морају учествовати у овом процесу, с обзиром на могућност одвијања комуникације у оба смера, истовремено. За успостављање конекције било је неопходно усаглашавање обе стране. Међутим, када је реч о затварању исте довољно је само иницирање неког од учесника. Полузатварање је процес који је омогућава да једна страна затвори конекцију, док други смер остаје отворен.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био доцент др Жељко Вуковић.

2.1. Предности *TCP* протокола

Циљ рада је да идентификујемо предности и мане *TCP* протокола, односно ситуације у којима је овај протокол прикладно решење за проблеме транспортног нивоа.

Једна од најзначајнијих особина јесте да *TCP* протокол има механизме за гаранцију испоруке података, као и детекције и корекције грешака приликом преноса. Један од начина је подржан на нивоу самог сегмента, коришћењем *TCP* checksum поља у оквиру заглавља. Други начин се односи на постављање *ACK flag*-а у заглављу сегмента и слање таквог сегмента, односно поруке потврде. Овде заправо користимо механизам потврђивања пријема података и то је још један од начина контроле грешака.

Наравно, неретко се дешавају и ситуације да се оваква порука потврде, енкапсулирана у *IP* пакет, загуби у мрежи. Како би се избегло потенцијално непотребно кашњење и чекање потврде, поставља се време у ком се очекује пристизање сегмента са постављеним *ACK flag*-ом. За ову сврху се користи *RTO* (*Retransmission Time-Out*) тајмер.

Дешавају се и ситуације да изгубљени пакет ипак стигне на пријемну страну, а да је у међувремено пакет опет послат. Тада на одредишту имамо дупликате, али се одбацује онај који је накнадно пристигао.

Још једна од битних особина протокола јесте да се гарантује испорука пакета у редоследу који дефинише пошиљалац. *TCP* протокол је оријентисан на ток података и води рачуна о реконструкцији редоследа сегмената на пријемној страни. Тако нам, као што смо претходно навели, омогућава да на одредишту прихватамо сегменте у редоследу у ком су и послати.

Током комуникације, неретко долазимо у ситуацију да страна која шаље и пријемна страна не могу да усагласе брзину слања, односно примања пакета. У циљу спречавања таквог загушења мреже, *TCP* протокол нуди механизам контроле протока који се назива *sliding window*. Идеја овог механизма јесте да се комуникационе стране договоре о величини сегмента, односно количини података коју ће у једној итерацији размењивати. То је омогућено постављањем поља, у оквиру сегмента, који се назива *Window Size*.

2.2. Мане *TCP* протокола

Сврха овог потпоглавља је да наведемо све особине које су мане *TCP* протокола, како бисмо касније могли анализирати те мане и решења која имплементирају други протоколи.

Сам почетак комуникације је приметно спорији у поређењу са брзином даље комуникације, услед неопходног успостављања конекције које претходи преносу података. Процес успостављања конекције успорава саму мрежу.

Континуално слање података и њихово преузимање у редоследу у ком су и послате, представља предност, али иману *TCP* протокола. У одређеним ситуацијама, губљење једног пакета неће бити уочено на пријемној страни, тачније неће утицати на жељени резултат.

Уколико се користи у оквиру *LAN* (*Local Area Network*) мреже може се десити да не искористимо погодности протокола јер је дизајниран и оптимизован пре свега за *WAN* (*Wide Area Networks*) мрежу. Такође треба напоменути да *TCP* користи *HTTP/2*.

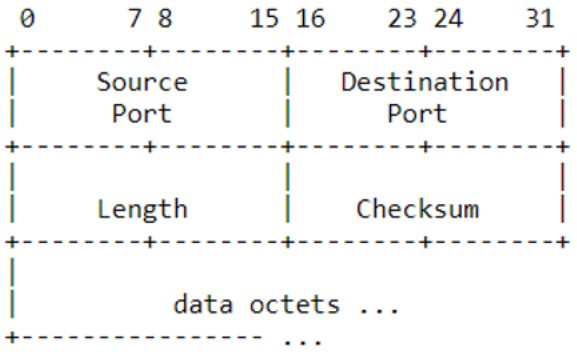
3. ДРУГИ ПРОТОКОЛИ ЧЕТВРТОГ НИВОА

Како смо у претходном поглављу побројали основне особине, предности и мане *TCP* протокола, сада ћемо више пажње посветити протоколима четвртог нивоа који се могу сматрати заменама за *TCP*.

3.1. *UDP* протокол

User Datagram Protocol (UDP) је настао 1980. године, а формално дефинисан и објављен у документу RFC 768 *[2]*. *TCP* и *UDP* су протоколи који се налазе на транспортном нивоу *OSI* модела. Протоколи нису дијаметрално различити, али одређене особине су од круцијалног значаја за један, односно други протокол чиме дефинишу његову сврху.

Пакети који се размењују *UDP* протоколом називамо *UDP datagram*. На слици 2 можемо погледати структуру једног таквог пакета. Посматрајући у односу на *TCP segment*, можемо приметити да је структура једноставнија.



Слика 2. *Формат заглавља UDP datagram-а [2]*

3.1.1. Разлике *TCP* и *UDP* протокола

TCP је *connection-oriented* протокол, који омогућава комуникацију између пошиљаоца и примаоца тек по успостави везе. Док је *UDP connectionless* протокол, што би значило да се не захтева успостава везе пре него што комуникација почне.

Битна особина коју треба нагласити јесте да *UDP* протокол не захтева успоставу везе и не гарантује испоруку свих пакета који су послати. Што се *TCP*-а тиче, он гарантује испоруку свих пакета, али захтева и успоставу везе пре почетка слања података. *UDP* не гарантује испоруку у тачно одређеном редоследу, нити омогућава превенцију од загушења тока података, што *TCP* омогућава.

Што се примене протокола тиче, *UDP* користимо у ситуацијама попут клијент-сервер конфигурације, где клијент шаље упит и очекује брз и кратак одговор. Поред тога, овај протокол интензивно се користи у *real-time* мултимедијалним апликацијама. Насупрот томе, *TCP* протокол је своју примену нашао у

ситуацијама у којима нам је неопходна поуздана комуникација, без губитка података.

3.2. Друге замене за *TCP* протокол

Осим набројана два најраспрострањенија протокола транспортног нивоа – TCP и UDP, временом су настали протоколи који покушавају да реше одређене проблеме специфичне за одређена поља примене и начине преноса, за које ни један од ова два постојећа протокола не представља адекватно решење.

Datagram Congestion Control Protocol (*DCCP*) је протокол транспортног нивоа *OSI* модела објављен 2006. године [3]. Имплементирано је поуздано успостављање, прекид везе, експлицитно обавештење о загушењу (*ECN*), контрола загушења и *feature nagotiation*.

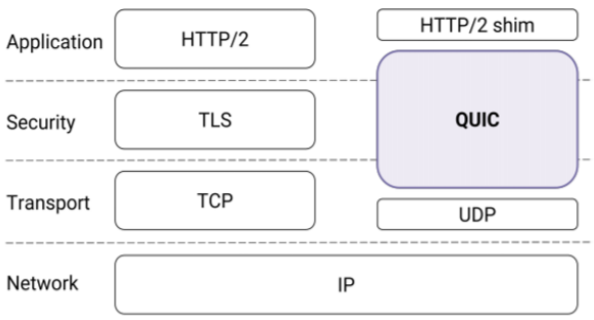
Поред *DCCP*-а, појавили су се и *Stream Control Transmission Protocol* (*SCTP*) 2007. године [4], *MultiPath TCP* (*MPTCP*) који се појавио у Јануару 2013. год [5].

QUIC протокол је комуникациони протокол развијен од компаније *Google*, који настоји да превазиђе мане данашњих комуникационих протокола и устали се као доминантан комуникациони протокол. Почетак развоја био је 2012. године, док је формално објављен тек маја 2021. године [6].

4. *QUIC* ПРОТОКОЛ

Посматрано из угла замене за *TCP* протокол, можемо констатовати да *QUIC* доста обећава. У прилог томе говори и чињеница да иза овог протокола стоји један гигант као што је *Google*. На серверској страни QUIC је имплементиран у светским гигантима попут *Google*, *Facebook*, *Microsoft*, *Apple*, итд. Текуће године водећи европски интернет провајдери објавили су информације да 20% њихових пакета иде преко QUIC-а.

Структура овог протокола нам омогућава да заменимо већину традиционалног HTTP стека, где се под тим подразумева HTTP/2, TLS и TCP приказан са леве стране на слици 3. Саобраћај је енкриптован, подаци су шифровани и обављена је превенција од измене истих. Комуникација подразумева енкриптовани *handshake* што ће нам омогућити да смањимо кашњење користећи унапред познате креденцијале сервера при поновној комуникацији. Тиме смо практично уклонили непотребно „представљање“ при свакој наредној комуникацији са истим сервером.



Слика 3. *Структура QUIC протокола [7]*

Како је QUIC протокол развијен скоро 40 година након TCP протокола, тако је у могућности да анализом претходних протокола и употребом савременије технологије развије напреднији вид комуникације.

Главне карактеристике протокола су:

- смањење кашњења,
- реализација комуникације кроз токове података,
- реакција на губитак података (пакета) и
- контрола загушења,
- безбедност и
- могућност одржања сесије и поред промене иницијалне адресе

Three-way handshake постао је оптерећавајући фактор који умногоме утиче на брзину успостављања конекције, а потом може и да проузрокује кашњење. Тако је QUIC нашао решење да се *handshake* не обавља у три етапе као раније, већ у два корака, док уколико је већ претходно успостављена конекција између клијента и сервера, да то буде само један корак.

Поред тога, *QUIC* обезбеђује комуникацију кроз випе токова података, чиме се повећава концептуална независност. Тиме је обезбеђено да семантички различити подаци путују кроз исту конекцију. Заправо, *QUIC* ће креирати неколико токова података преко исте *UDP* конекције. За разлику од *TCP*-а, који секвенцијално доставља пакете и нема могућност да избегне *head-of-line blocking*[1], QUIC осигурава да изгубљени *UDP* пакет може једино утицати на ток података који преноси тај пакет, чиме не може доћи до загушења. Веома важна особина токова је да се они идентификују путем *stream ID*. Оно што омогућава креирање више токова је то што су подаци енкапсулирани у један или више фрејмова, при чему један *QUIC* пакет може да садржи неколико фрејмова који су везани за токове података.

Како је реакција на губитак података код *TCP*-а доста поуздана, али спора, тако *QUIC* омогућава унапређење и овог сегмента. За разлику од *TCP*-а, *QUIC* ће започети опоравак од губитка пакета, одмах након што се исти изгуби, где се епоха опоравка завршава онда када се било који пакет, који је послат након тога, прихвати на другој страни [8]. Опоравак се може реализовати:

- поновним покушајем преношења истих података,
- слањем измењеног frame-а или
- одбацивањем frame-а

Како је *QUIC* базиран на *UDP*-у који нема имплементирану контролу загушења, стога он то ради на апликативном нивоу. Није везан ни за један конкретан алгоритам контроле загушења, већ има интрефејс који је *plug*-абилан где притом дозвољава различите имплементације. Подразумевани контролер загушења је *Cubic*. Као што смо у претходном потпоглављу дефинисали, *QUIC* је у стању да детектује губитак пакета, а ту информацију може да искористи да прилагоди контролер загушења. Сваки контролер задужен је за једну путању, где притом нема интерференције између контролера на различитим путањама.

QUIC примењује екрипцију на транспортни ниво. Цео *UDP* садржај који се преноси је аутентификован и

[1] head-of-line blocking - феномен је који ограничава перформансе и јавља се када су два независна захтева непотребно блокирана због ограничења протокола

обављена је превенција од потенцијалне измене и практично сви подаци који се преносе су енкриптовани. Делови пакета који нису енкриптовани, битни су нам за рутирање или декриптовање пакета. Ту спадају *Flags, Connection ID, Version Number, Diversification Nonce* и *Packet Number.*

QUIC протокол, као што је претходно поменуто, за сваку конекцију чува посебну вредност – *Connection ID.* Овај идентификатор јединствено идентификује сваку конекцију која је остварена овим протоколом. Чиме имамо могућност да наставимо отворену сесију са једне адресе, на другој адреси.

4.1. Демонстрација истраживања

За демонстрацију рада *QUIC* протокола искоришћена је библиотека *LSQUIC* [9], како бисмо на једноставнијем примеру показали на који начин се подаци размењују применом поменутог протокола. Ова библиотека представља *open-source* имплементацију *QUIC*-а.

Било је неопходно *setup*-овати окружење. Подигнута је виртуелна машина са *Linux* оперативним системом. Након постављања виртуелне машине, потребно је било креирати погодно окружење за компајлирање и стартовање библиотеке.

Након стартовања библиотеке, прешли смо на практични део. Подигнута је једноставна апликација за потребе задатка. Апликација је *deploy*-ована и такође покренута. Следећи задатак је био да размењујемо податке са апликацијом користећи *QUIC* протокол.

Ограничавајућа околност при изради задатка било је то што је протокол веома млад и стога нема довољно података и информација које би биле од користи при практичном делу. Тако смо се при имплементацији морали ослонити на *RFC* документ [6] и документацију за *LSQUIC* библиотеку [9].

5. ЗАКЉУЧАК

Циљ рада био је да се осврнемо на *TCP* протокол, да сагледамо његове карактеристике и да истражимо алтернативе за исти. Акценат рада стављен је на *QUIC* протокол, стога је сврха задатка упознавање и демонстрација рада са истим. Дефиниција и обрада начина функционисања тог протокола дају нам релевантне чињенице које га истичу у односу на *TCP* протокол.

Приказана библиотека *LSQUIC* омогућава једноставну имплементацију клијентске стране софтверских решења за које је потребно подржати комуникацију QUIC протоколом.

У даљем току истраживања неопходно је идентификовати и анализирати библиотеке које омогућавају имплементацију и серверске стране. Од значаја може бити и анализа и поређење других у раду побројаних транспортних протокола виђених као алтернатива за TCP и UDP.

6. ЛИТЕРАТУРА

[1] Fu-Hau Hsu, Yan-Ling Hwang, Cheng-Yu Tsai, Wei-Tai Cai, Chia-Hao Lee, KaiWei Chang, “TRAP: A Three-Way Handshake Server for TCP Connection Establishment”, Новембар 2016.

[2] J.Postel, “RFC 768”

[3] E. Kohler, M. Handley, S. Floyd, “RFC 4340”

[4] R. Stewart, “RFC 4960”

[5] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, “RFC 6826”

[6] J. Iyengar, M. Thomson, “RFC 9000”

[7] Adam Langley, Alistair Riddoch, Alyssa Wilk, Antonio Vicente, Charles Krasic, Dan Zhang, Fan Yang, Fedor Kouranov, Ian Swett, Janardhan Iyengar, Jeff Bailey, Jeremy Dorfman, Jim Roskind, Joanna Kulik, Patrik Westin, Raman Tenneti, Robbie Shade, Ryan Hamilton, Victor Vasiliev, Wan-Teh Chang, Zhongyi Shi, “The QUIC Transport Protocol: Design and Internet-Scale Deployment”, Август 2017.

[8] J. Iyengar, M. Thomson, “RFC 9002”

[9] https://lsquic.readthedocs.io/en/latest/ (приступљено у августу 2021.)

Кратка биографија:

Александар Бекер рођен је у Новом Саду 1997. године. Мастер рад на Факултету техничких наука из области Електротехнике и рачунарства – Примењене рачунарске науке и информатика одбранио је 2021. године.

контакт: acabeker@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.41
DOI: https://doi.org/10.24867/15BE06Isic

MIGRACIJA SCADA SISTEMA U CLOUD OKRUŽENJE

MIGRATION OF SCADA SYSTEMS TO THE CLOUD ENVIRONMENT

Vuk Isić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – PRIMENJENO SOFTVERSKO INŽENJERSTVO

Kratak sadržaj – *Ovaj rad se bavi razmatranjima vezanim za migraciju SCADA sistema na Cloud. Problem migracije SCADA sistema na Cloud je kompleksan problem. Za ovakav problem ne postoji jednostavno rešenje, niti rešenje koje je već negde primenjeno, pa se može primeniti i na ovakav problem.*

Ključne reči: *SCADA, Cloud, Migracija, Mikroservisna arhitektura, Web, Nadzor, Upravljanje, Akvizicija.*

Abstract – *Migration of SCADA systems to the Cloud environment is presented in this paper. The problem of migrating SCADA systems to the Cloud is a complex problem. There is no simple solution that has already been applied somewhere, so it can be applied to this problem as well.*

Keywords: *SCADA, Cloud, Migration, Microservices, Web, Control, Commanding, Acquisition.*

1. UVOD

Konstantan napredak tehnike i razvoj novih tehnologija zajedno utiču na promene u svim sferama softverske industrije. Između brojnih dostupnih tehnologija, postoji jedna koja se posebno ističe – ***Cloud***. ***Cloud*** je drastično izmenio softversku industriju. Omogućio je stvaranje novih arhitektura i klasa aplikacija i uticao je na način na koji razmišljamo o razvoju i primeni softvera. Kako ***Cloud*** računarstvo utiče na gotovo sve softverske sisteme, pa tako se sve češće pokreće diskusija o primeni ***Cloud*** računarstva u infrastrukturnim sistemima sa kritičnom misijom.

Infrastrukturni sistemi su sistemi od ključnog značaja za rad industrije i socijalni kvalitet života. Upravljanje ovakvim sistemima je veoma kompleksno zbog velikog broja parametara koji utiču na njihov rad i uticaja na *„korisnike“* ovakvih sistema [1].

Iz prethodno navedenih razloga, primena novih tehnologija u infrastrukturnim sistemima je nešto sporija o odnosu na ostale sisteme i samim promenama se pristupa na veoma ozbiljan način uzimajući u obzir moguće rizike i uticaje. Zbog performansi i nivoa bezbednosti koje ovakvi sistemi treba da postignu, veoma često se primenjuju lokalna rešenja. Međutim, kako se ***Cloud*** računarstvo brzo razvija, postaje očigledno da će u nekom trenutku u budućnosti, kada se za to steknu adekvatni uslovi, i infrastrukturni sistemi izvršiti migraciju na ***Cloud***.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Darko Čapko, vanr. prof.

Jedan od vitalnih delova svakog infrastrukturnog sistema jeste ***SCADA*** sistem, koji u osnovi omogućuje nadzor i upravljanje sistemom u režimu realnog vremena sa udaljene lokacije. Kompleksnost ***SCADA*** sistema se ogleda u zagarantovanom vremenu odziva *(eng. Real-time)* sistema koje je potrebno obezbediti.

Glavni zadatak ovog rada jeste razmatranje migracije ***SCADA*** sistema na ***Cloud***. Sama razmatranja moraju uključiti analizu i validaciju mogućih rešenja, prikaz alternativa, prikaz mogućnosti, prednosti i mana ove migracije.

2. TEORIJSKE OSNOVE

2.1. SCADA

SCADA*(eng. Supervisory Control And Data Acquisition)* predstavlja sistem za nadzor i upravljanje realnim sistemima. Spada u klasu računarskih sistema za rad u realnom vremenu – sistemi koji treba da obezbede adekvatan *(zagarantovan)* odziv na promene stanja kontrolisanog objekta [2].
Osnovne funkcije ***SCADA*** sistema su:

- **Prikupljanje podataka** – Tok podataka od senzora ka glavnoj upravljačkoj stanici [2]. Podaci se mogu klasifikovati u dve kategorije: Indikacije statusa i Merene vrednosti. Postoje dva načina za prikupljanje podataka: Periodično prozivanje i Izveštavanje po izuzetku**.**
- **Nadzor** – nadzor prikupljenih podataka u odnosu na definisana normalna stanja i ograničenja. Ukoliko dođe do narušavanja ograničenja, generišu se alarmi.
- **Upravljanje** – Tok podataka od glavne upravljačke stanice ka aktuatorima [2] predstavlja upravljanje. Upravljanje može biti manuelno ili automatsko, u skladu sa zahtevima samog postrojenja kojim se upravlja.

U ***SCADA*** sistemima se primenjuju različiti protokoli za komunikaciju sa uređajima u polju koji su bazirani na ***TCP/IP*** komunikacionom steku. Najpoznatiji ***SCADA*** protokoli su: ***Modbus***, ***DNP3*** i ***IEC101/104***.

Kako ***SCADA*** sistemi često upravljaju kritičnim infrastrukturnim sistemima, ***SCADA*** softver se klasifikuje kao softver sa kritičnom misijom.

2.2. Cloud računarstvo

Za pojam ***Cloud*** računarstva *(eng. Cloud computing)* postoji veliki broj definicija u zavisnosti od ugla posmatranja, ali se generalno može reći da ***Cloud*** predstavlja način za isporuku ***IT*** resursa na zahtev putem interneta [3].

Glavne karakteristike ***Cloud**-a* su: Performanse, Lokacijska nezavisnost, Pouzdanost, Skalabilnost, Dostupnost i Virtualizacija.

Tipovi ***Cloud**-a*:

- ***Public Cloud*** – Javno dostupan svima putem standardnog interneta. *Hostovan* za različite korisnike i različite namene [3].
- ***Private Cloud*** – Lokalno *hostovan* od strane određene kompanije. Ograničeni broj korisnika može da mu pristupi [3].
- ***Hybrid Cloud*** – Kombinacija prethodna dva – virtuelni privatni ***Cloud***. Omogućuje korišćenje servisa u ***Public Cloud***-u, ali kontroliše ko može da im pristupi [3].

Nivoi usluga na ***Cloud**-u*:

- ***IaaS*** *(eng. Infrastructure as a Service)* – Osnovni hardverski resursi (računarski, skladišni i mrežni).
- ***PaaS*** *(eng. Platform as a Service)* – Osnovni hardverski i softverski resursi *(na PaaS-u gradimo SaaS).*
- ***SaaS*** *(eng. Software as a Service)* – Odgovornost pružaoca usluga je celokupna isporuka servisa. Softver se isporučuje kao servis.

3. OPIS PROBLEMA

Problem migracije ***SCADA*** sistema na ***Cloud*** je kompleksan problem. Za ovakav problem ne postoji jednostavno rešenje, niti rešenje koje je već negde primenjeno, pa se može primeniti i na ovakav problem. Zbog toga je sam problem migracije ***SCADA*** sistema na ***Cloud*** veoma aktuelan problem.

Osnovni problem jeste fundamentalna suprotnost između ***SCADA*** sistema i ***Cloud*** računarstva.

SCADA sistemi su ***real-time*** sistemi – sistemi sa zagarantovanim vremenom odziva, koji su veoma osetljivi na vremenska kašnjenja, brzinu procesiranja i greške u komunikaciji. Dodatno, samo postrojenje se može sastojati iz više hiljada tačaka koje je neophodno nadzirati i kojima je potrebno upravljati.

S druge strane, ***Cloud*** predstavlja način za pristup objedinjenim ***IT*** resursima putem interneta. ***Cloud*** nudi prednosti poput skalabilnosti, dostupnosti, pouzdanosti i smanjenja troškova. Međutim ***Cloud*** sa sobom nosi i odgovarajuće probleme u pogledu performansi, bezbednosti i privatnosti. Ključno za upotrebu ***Cloud**-a* jeste razumevanje usluga u ***Cloud**-u*.

SCADA sistemi su vremenom evoluirali od klasičnih centralizovanih rešenja do distribuiranih rešenja. Sledeći logični korak jeste ***Cloud***. Na ***Cloud**-u* se koriste i pristupa se servisima. Međutim sam pristup servisu na ***Cloud**-u* putem interneta unosi promenljivo kašnjenje. Dodatno, nijedan ***Cloud*** provajder trenutno ne nudi komponente za rad u realnom vremenu, niti garantuje za vreme odziva servisa – ***SCADA*** sistemima je upravo vreme odziva ključna karakteristika. Generalno treba migrirati ***real-time*** sistem na platformu / okruženje koje nije ***real-time*** kao što je ***Cloud***.

Problem migracije se mora razmotriti pre svega sa tehničkog stanovišta, ali treba uzeti u obzir i druge faktore. Trenutno ne postoji konkretno rešenje za ovu migraciju koje je u širokoj primeni. Ključ za uspešnu migraciju leži u razumevanju obe strane, i ***SCADA*** sistema i ***Cloud**-a*, razumevanju potrebe za migracijom i definisanju odgovarajućeg optimalnog rešenja za obe strane.

4. ARHITEKTURA

Arhitektura komponenata ***SCADA*** sistema je mikroservisna arhitektura i prikazana je na slici 1. Mikroservisna arhitektura je evolucija servisno-orijentisane arhitekture sa fokusom da svaki servis izvršava tačno jednu biznis logiku.

Komponente sistema su sledeće:

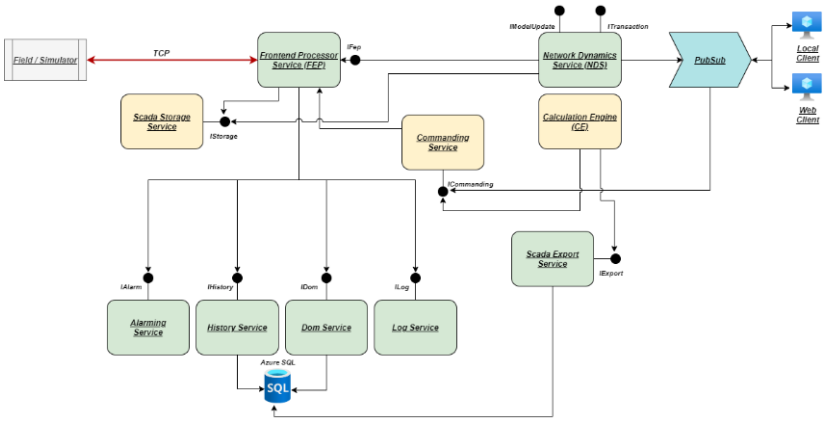
- ***Frontend Processor Service*** – Komponenta koja se koristi za komunikaciju sa uređajima u polju.
- ***Network Dynamics Service*** – Komponenta zadužena za rad sa modelom postrojenja.
- ***Scada Storage Service*** – Komponenta zadužena za skladištenje podataka o merenjima i modelu.
- ***Commanding Service*** – Komponenta koja primi ***SCADA*** komande, obradi ih i pošalje na izvršenje.
- ***Calculation Engine*** – Komponenta zadužena za proračune i automatsko upravljanje postrojenjem.
- ***Alarming Service*** – Komponenta zadužena za obradu i generisanje alarma za sva merenja.
- ***History Service*** – Komponenta zadužena za zapis istorije merenja.
- ***Dom Service*** – Komponenta zadužena za nadzor stanja opreme.
- ***Log Service*** – Komponenta zadužena za zapis ***log*** podataka celog ***SCADA*** sistema.
- ***Scada Export Service*** – Komponenta koja omogućuje pristup podacima ***SCADA*** sistema.
- ***Pub Sub*** – Komponenta za komunikaciju između ***SCADA*** sistema i klijentskih ***HMI*** aplikacija.
- ***Local / Web Client*** – Klijentske ***HMI*** aplikacije.
- ***Simulator*** – Komponenta koja simulira realni procesni kontroler.

Postoje dva tipa servisa:

- *Stateless servisi* – servisi koji nemaju neko interno specifično stanje *(obojeni zelenom bojom na slici)*. Servis koristi druge resurse kako bi izvršio svoj zadatak. Instanca servisa se kreira na zahtev.
- *Stateful servisi* – servisi koji poseduju neko interno specifično stanje *(obojeni žutom bojom na slici)*. Samo stanje se replicira na sve instance tog servisa.

Komunikacija u sistemu može biti realizovana na razne načine, jedini izuzetak jeste komunikacija sa simulatorom. Ona se realizuje putem ***TCP**-a* jer je većina ***SCADA*** protokola orijentisana na upotrebu ***TCP**-a*. Ostatak sistema može koristiti različite tehnologije za komunikaciju.

Važno je napomenuti da je ovakva arhitektura ***SCADA*** sistema uopštena *(referentna)* i da se može razlikovati u zavisnosti od konkretne primene u konkretnom sistemu. Cilj ove arhitekture je da obuhvati sve bitne komponente ***SCADA*** sistema.



Slika 1. *Arhitektura SCADA sistema*

5. ANALIZA MOGUĆIH REŠENJA

5.1 Public Cloud rešenje

Glavna odlika ovog rešenja jeste uniformnost – sve komponente / servisi se nalaze na ***Public Cloud-u*** - nema tranzicije između različitih okruženja.

Komunikacija između komponenti / servisa može biti realizovana na različite načine. Takođe, može se koristiti i otvorena infrastruktura za komunikaciju putem interneta.

Glavna prednost ovakvog rešenja jeste cena. ***Public Cloud*** je generalno jeftiniji od ***Private*** ili ***Hybrid Cloud****-a*.

Glavni nedostaci ovog rešenja su bezbednost, privatnost i performanse. ***Public Cloud*** je napravljen tako da je dostupan svima i da se može koristiti za različite primene, što dovodi do problema izolacije podataka. Nepredvidivo kašnjenje utiče na performanse vezane za pristup servisima.

Primena ovog rešenja je dosta upitna zbog navedenih problema. Ako se obezbedi odgovarajući nivo bezbednosti, ovakvo rešenje se može primeniti na ***Soft SCADA*** sisteme koji se odlikuju ublaženim zahtevima vezanim za vreme odziva.

5.2 Private Cloud rešenje

Glavna odlika ovog rešenja, ponovo, jeste uniformnost – sve komponente / servisi se nalaze na ***Private Cloud-u*** - nema tranzicije između različitih okruženja.

Komunikacija između komponenti / servisa može biti realizovana na različite načine. Dodatno, mogu se uključiti i privatne ili neke specifične komunikacione veza kojima su inače ***SCADA*** sistemi skloni.

Glavne prednosti ovog rešenja jesu performanse i bezbednost.
Kako je ***Private Cloud*** hostovan za tačno određenog klijenta i u tačno definisane svrhe sledi da je izolacija podataka potpuna. Sama privatna komunikaciona infrastruktura unapređuju performanse jer su komunikaciona kašnjenja manja, a komunikacioni linkovi su manje opterećeni.

Glavni nedostaci ovog rešenja su cena (***Private Cloud*** je generalno skuplji od ***Public*** i ***Hybrid Cloud****-a*) i moguća ograničena skalabilnost ***Private Cloud****-a*.

Rešenje migracije ***SCADA*** sistema u ***Private Cloud*** je generalno rešenje koje se najviše preporučuje, a razlozi za to su performanse, bezbednost i privatnost.

5.3 Hybrid Cloud rešenje

Kako rešenja u ***Public Cloud-u*** i ***Private Cloud-u*** imaju različite prednosti i mane, moguće rešenje može biti integracija ova dva rešenja - upotreba ***Hybrid Cloud-a***.

Glavna odlika ovog rešenja jeste, integracija različitih okruženja. Da bi se ovakvo rešenje moglo realizovati, mora se postaviti jasna granica između najkritičnijih komponenti i ostalih.

U ovom rešenju najkritičnije komponente su komponente za direktnu interakciju sa postrojenjem ***(Frontend Processor***, ***Scada Storage Service*** i ***Commanding Service)***. Ove komponente su odabrane za primenu u ***Private Cloud****-u* jer zahtevaju visoke performanse.

Ostale komponente su takođe kritične, ali sve one direktno zavise od performansi komponenti koje se nalaze u ***Private Cloud****-u* pa su iz tog razloga one odvojene za primenu u ***Public Cloud****-u*.

Komunikacija između komponenti može biti realizovana na različite načine odvojeno između komponenti u ***Public*** i ***Private Cloud****-u*.

Glavna prednost ovog rešenja jeste to što se kombinuju prednosti prethodna dva rešenja. Rešenje primenom ***Hybrid Cloud****-a* je jeftinije od rešenja na ***Private Cloud****-u* ali skuplje od rešenja na ***Public Cloud****-u*, a pri tome se performanse, iako neznatno lošije u globalu, ne razlikuju puno od rešenja na ***Private Cloud****-u*. Nivo bezbednosti je unapređen u odnosu na rešenje u ***Public Cloud****-u*, a skalabilnost je unapređena u odnosu na rešenje u ***Private Cloud-u***.

Glavni nedostatak ovog rešenja jeste kompleksna arhitektura. Granice između delova sistema u različitim okruženjima moraju biti jasno postavljene. Komunikacija između okruženja mora biti pažljivo planirana kako ne bi postala *„usko grlo“*.

Rešenje upotrebom ***Hybrid Cloud-a*** je rešenje koje se nalazi na polovini puta između ***Public Cloud*** rešenja i ***Private Cloud*** rešenja. Generalno nudi kompromis između performansi i cene.

5.4 Hibridno Rešenje

Prethodna rešenja prikazala su mogućnosti migracije ***SCADA*** sistema na ***Cloud***, kada se migrira ceo ***SCADA*** sistem. Međutim, postavlja se jedno legitimno pitanje:

Zašto ne bismo jedan deo ***SCADA*** sistema ostavili da radi u lokalu, a ostatak sistema migrirali na ***Cloud***?

Glavna odlika ovog rešenja jeste integracija lokalnog sistema sa ***Cloud***-*om*. Ponovo se pojavljuje potreba da se jasno definišu granice između heterogenih okruženja.

Najkritičnije komponente vezane za direktnu interakciju sa postrojenjem su odvojene u posebnu celinu koja se izvršava u lokalu. Te komponente ponovo su ***Frontend Processor***, ***Scada Storage Service*** i ***Commanding Service***. Na ovaj način ovaj deo sistema ima mogućnost da ostvari najbolje performanse vezane za interakciju sa postrojenjem.

Ostale komponente su takođe kritične ali sve one direktno zavise od performansi komponenti koje se nalaze u lokalu pa su one odvojene za primenu u ***Cloud***-*u*.

Jedan od glavnih izbora koje je potrebno napraviti, pored izbora šta će biti u lokalu, a šta će biti na ***Cloud***-*u*, jeste koji tip ***Cloud***-*a* će se koristiti za komponente na ***Cloud***-*u*. Jasno je da u pogledu performansi i bezbednosti bolji izbor predstavlja ***Private Cloud***.

U pogledu komunikacija pre svega se mora razmatrati komunikacija između lokalnog sistema i ***Cloud-a***. S obzirom da se za ovu komunikaciju najčešće koristi internet, mora se posebno povesti računa o bezbednosti tih komunikacija. Komunikacija između komponenti na ***Cloud-u*** može biti realizovana na različite načine.

Glavne prednosti ovakvog sistema jesu performanse. Najkritičniji deo sistema je u lokalu i odziv ovog dela sistema je ***real-time***. Deo sistema na ***Cloud***-*u* se koristi kako bi se obezbedile mogućnosti masovnog paralelnog procesiranja i skladištenja podataka.

Glavni nedostatak ovog sistema jeste, ponovo, kompleksna arhitektura. Granice između delova sistema moraju biti jasno postavljene. Komunikacija između okruženja mora biti pažljivo planirana kako ne bi postala *„usko grlo“* između dva okruženja.

Ovo hibridno rešenje može imati dve uloge, može predstavljati inicijalno rešenje za migraciju na ***Cloud***, ali takođe ovo rešenje može biti i konačno rešenje za migraciju.

6. PRIMER

Primer predstavlja migraciju ***MasterSCADA*** projekta [4] na ***Cloud***. U prvoj fazi razvoja, implementiran je distribuirani ***SCADA*** sistem. U drugoj fazi, pristupilo se migraciji ***SCADA*** sistema na ***Public Cloud***.

Prilikom migracije ***SCADA*** sistema na ***Public Cloud***, primećeno je da su, u pogledu performansi, vremena odziva sistema povećana, delimično iz razloga upotrebe ***Public Cloud***-*a*, kao i da su otvorena nova pitanja u smislu bezbednosti (Tabela 1. *Poređenje performansi*).

Tabela 1. *Poređenje performansi*

Rešenje	Akvizicija	Izvršenje komande	Ažuriranje GUI-a	Ukupno vreme odziva
Lokalno	2 s	~5 s	5 s	~12 s
Public Cloud	5 s	~11 s	10 s	~26 s

7. ZAKLJUČAK

SCADA sistemi su evoluirali od monolitnih – centralizovanih rešenja ka distribuiranim rešenjima. Sledeći, logični korak u razvoju, jeste migracija ***SCADA*** sistema na ***Cloud***. Međutim, migracija ***SCADA*** sistema na ***Cloud*** predstavlja kompleksan problem, za koji ne postoji jednostavno rešenje.

U pogledu predloženih rešenja, rešenje u ***Private Cloud***-*u* se izdvaja po performansama i bezbednosti, ali karakteristika takvog rešenja je visoka cena i moguća ograničena skalabilnost. S druge strane se nalazi rešenje u ***Public Cloud***-*u* koje je u pogledu cene jeftinije i u pogledu skalabilnosti bolje, ali postoje problemi sa performansama i privatnošću. Kao kompromis između rešenja u ***Public*** i ***Private Cloud-u***, predlažu se i hibridna rešenja. Prvo hibridno rešenje se zasniva na upotrebi ***Hybrid Cloud*** okruženja i nudi kompromis između cene i performansi i predstavlja jedno veoma primenljivo rešenje. Drugo hibridno rešenje zasniva se na tome da se najkritičnije komponente sistema ostave u lokalu, dok se procesiranje i obrada podataka izvršavaju na ***Private Cloud***-*u*. Na taj način se mogu iskoristiti prednosti ***Cloud***-*a* poput skalabilnosti, dostupnosti i pouzdanosti. Ovo, drugo rešenje, je kao i prvo hibridno rešenje dosta primenljivo, međutim u oba rešenja postoji problem kompleksne arhitekture.

U pogledu svih mogućih predloženih rešenja, predlaže se primena rešenja u ***Private Cloud***-*u* kao i primena hibridnih rešenja. U zavisnosti od konkretnog sistema treba izabrati jedno od tih rešenja u zavisnosti od potreba konkretnog sistema.

8. LITERATURA

[1] A. Selakov, „Napredni infrastrukturni sistemi“ - materijal sa predavanja, 2021.

[2] B. Atlagić, Softver sa kritičnim odzivom - Projektovanje SCADA sistema, Novi Sad: Fakultet tehničkih nauka u Novom Sadu, 2015.

[3] S. Vukmirović, Cloud Computing - Materijal sa predavanja, Novi Sad: Fakultet tehničkih nauka, 2021.

[4] V. Isić, S. Košutić, M. Janković, Ž. Matović i D. Novaković, „Master Projekat - MasterSCADA,“ 2021.

Kratka biografija:

Vuk Isić rođen je 26.10.1997. godine u Valjevu. Diplomirao je na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Primenjeno softversko inženjerstvo 2020. godine.

Kontakt: vukisic97@gmail.com.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/15BE07Vuckovic

AUTOMATIZOVANO TESTIRANJE KORISNOSTI VEB APLIKACIJA POMOĆU JASMIN OKVIRA

AUTOMATIC TESTING THE USEFULNESS OF WEB APPLICATIONS USING THE JASMINE FRAMEWORK

Marko Vučković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – Računarstvo i automatika

Kratak sadržaj – *Rad se sastoji iz dva dela. Prvi deo su osnove testiranja, a zatim i primena istog uz pomoć Jasmin okvira. Jasmin je jedan od alata koji se mogu primeniti za testiranje veb aplikacija. Prikazane su osnovne funckionalnosti koje Jasmin sa sobom donosi*

Ključne reči: *Jasmin, testiranje, veb aplikacije*

Abstract – *This paper consists of two parts. The first part is the basics of testing, and second part is testing of web applications using help of Jasmine framework. Jasmine is one of tools that can be used for testing web applications. The basic functionalities that Jasmine brings are presented in the paper.*

Keywords: *Jasmine, testing, web applications.*

1. UVOD

U poslednjoj deceniji svedoci smo da je softver pojam koji nas svakodnevno okružuje. Gde god da krenemo, bilo to u prodavnicu, na sport ili negde drugde, sve je postalo kompjuterizovano. Softverski sistemi postaju sve više važniji, kompleksniji i raste značaj njihovog kvaliteta. Samim tim, kontrola kvaliteta softvera je važna koliko i sama izrada rešenja. Propust može dovesti do ozbiljnjih problema, ukoliko se govori o oblastima poput medicine, ili vazduhoplostva, dok na drugoj strani može izazvati nepoverenje klijenata, a poverenje u softver je jedno od najbitnijih stavki kada se softver isporuči klijentu. Testiranje softvera predstavlja samo jedan deo procesa kontrole kvaliteta softvera.

Različite metode koriste se za testiranje različitih slojeva softvera, a ono što je svima zajedničko jeste da se mogu izvoditi manuelno ili automatski korišćenjem posebnih alata. I jedan i drugi način imaju svoje prednosti i mane, ali se u poslednje vreme sve više teži automatizaciji jer se automatizacijom testiranja skraćuje vreme potrebno za testiranje.

Jasmine je okvir za automatizovano testiranje web aplikacija. Jedan je od najčešćih izbora stručnjaka koji se bave testiranjem softvera. Univerzalan je, lak za korišćenje, i podržava testiranje u većinih dostupnih pretraživača. U ovom radu su opisane osnovne funkcionalnosti koje dolaze uz Jasmin okvir.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Dragan Ivetić.

2. TESTIRANJE SOFTVERA

Testiranje softvera je proces koji se koristi da bi se utvrdila ispravnost, potpunost i kvalitet razvijenog softvera u potpunosti da utvrdi ispravnost računarskog softvera. Samo proces formalne verifikacije može da pokaže da u softveru nema grešaka. Testiranje softvera je način da se obezbedi manji broj grešaka, manji trošak održavanja i sveukupne cene softvera.

2.1. Osnovni pojmovi

Često se dešava u praksi da se određeni pojmovi pogrešno koriste i tumače. Koja je razlika između greške i defekta? Šta je incident? Kako bi se izbegle nedoumice, neophodno je na samom početku definisati osnovne pojmove. Greška jeste rezultat ljudske aktivnosti, bilo za vreme specifikacije programa ili tokom pisanja programa. Na primer, neka funkcionalnost može biti pogrešno specificirana, što će kasnije dovesti do pogrešne implementacije, ili se može napraviti neka greška u kodiranju, koja će dovesti do neispravnog rada programa. Posledica greške se naziva mana ili defekt – programu nešto nedostaje , ili ima funkciju koja se ne ponaša ispravno. Otkaz nastaje kada sistem nije u mogućnosti da obavi funkciju koju korisnik od njega zahteva, jer se aktivira i izvršava defektni kod. Kada postoji defekt u softveru, javlja se simptom kojim korisnik postaje svestan otkaza u sistemu.

Ovaj simptom se naziva incident (ili poremećaj). Testiranje je postupak izvršavanja softvera sa testovima. Testiranje može imati jedan od dva osnovna cilja: da se pronađu otkazi ili da se demonstrira ispravno ponašanje sistema. Testiranjem se mogu otkriti otkazi u sistemu, koje je kasnije potrebno ispraviti.

Test je skup ulaznih vrednosti, preduslova izvršavanja, očekivanih rezultata i stanja u kome sistem treba da ostane nakon završetka. Test se razvija sa ciljem da ispita određeno ponašanje programa, na primer da izvrši određenu putanju kroz program ili da verifikuje da li je određeni zahtev ispravno implementiran. Testiranje softvera se može definisati kao proces izvršavanja programa sa ciljem da se pronađu greške. Ukoliko posmatramo širi smisao, testiranje je proces koji se sastoji od statičkih i dinamičkih aktivnosti, sa ciljem da se odredi da li softver zadovoljava specificirane zahteve i otkriju defekti ili da se demostrira ispravan rad.

Testiranje softvera je deo procesa osiguravanja kvaliteta (engl. Quality Assurance), čiji je zadatak da se obezbedi isporuka kvalitetnog softvera u zahtevanom vremenskom roku. Test set se odnosi na skup svih testova koje je

potrebno izvršiti nad softverom. Testovi se ne biraju nasumično, već je neophodno uložiti trud u pažljivo planiranje i dizajn. Čitava jedna faza u procesu testiranja softvera je odvojena u ovu svrhu. Nasumično odabrani test nema nikakvog značaja ukoliko detektuje grešku koja je već pronađena nekim drugim testom. Testiranje svih mogućih kombinacija ulaznih podataka ili takozvano iscrpno testiranje je nemoguće ili u najboljem slučaju nepraktično, pošto svaki netrivijalni sistem ima veoma veliki domen ulaznih podataka. Ukupan broj testova u test setu ne garantuje da će i testiranje biti uspešno.

2.2. Proces testiranja softvera

Testiranje je proces određivanja nivoa kvaliteta softvera. Nivo kvaliteta softvera i zahtevi variraju od sistema do sistema, ali neke osnovne karakteristike koje svi oni treba da poseduju i koje se vrednuju su: pouzdanost, stabilnost, prenosivost, kompatibilnost i upotrebljivost. Testiranje je usko povezano sa pojmovima verifikacije i validacije softvera.

Verifikacija je proces procene softvera ili njegovih komponenti sa ciljem da se utvrdi da li proizvodi određenih faza razvoja zadovljavaju uslove sa početka tih faza. Ovim procesom se utvrđuje koliko je proizvod jedne faze kompatibilan sa proizvodom prethodnih faza i da li svi delovi sistema, pojedinačno i kao celina, rade ispravno. Validacija se definiše kao proces procene softvera ili njegovih komponenti, tokom ili na kraju njegovog razvoja, sa ciljem da se utvrdi da li zadovljava polazne zahteve. Cilj validacije jeste da odgovori na pitanje da li se razvija pravi softver, to jest, softver koji je u skladu sa zahtevima korisnika definisanim na početku procesa razvoja i da li je takav softver upotrebljiv za korisnika. Validaciju obavljaju testeri kroz izvršavanje test skriptova.

2.3. Osnovni principi testiranja

Postoji sedam osnovnih principa testiranja, koji su se izdvojili kroz nekoliko decenija prakse testiranja softvera, i koji važe za bilo koji tip softvera. Posmatraju se kao smernice koje bi trebalo ispoštovati u svakom projektu, jer su se kroz istoriju pokazali kao tačni.

Principi glase: testiranje pokazuje prisustvo defekata, iscrpno testiranje nije moguće, rano testiranje, grupisanje defekata, paradoks pesticida, testiranje zavisi od konteksta, odsustvo grešaka ne garantuje da sistem radi kako treba.

2.4. Stragetije testiranja softvera

U odnosu na način na koji se posmatra sistem koji se testira, postoje dva pristupa testiranju – tehnika bele kutije i tehnika crne. Metoda crne kutije je oblik testiranja softvera gde unutrašnja struktura, dizajn i implementacija softvera ili softverskih jedinica koje se testiraju nisu poznate testeru. Kako implementacija i struktura softvera nije poznata, softver se posmatra kao crna kutija po čemu je ovaj metod i dobio ime. Jedina poznata informacija koju tester ima za određivanje i dizajn testova je specifikacija zahteva programa.

Metodom crne kutije se mogu otkriti nepostojeće funkcionalnosti ili pogrešne implementacije u softveru, greške u ponašanju softvera kao i problemi sa performansama sistema. Tehnika crne kutije se može primeniti na svim nivoima testiranja (jediničnom, integracionom i sistemskom nivou). Uticaj nivoa testiranja na metodu crne kutije se ogleda samo u kompleksnosti izvršavanja ove metode. Strukturno testiranje, poznato kao metoda bele ili staklene kutije, je tehnika testiranja gde je testerima poznata implementacija softvera. Za razliku od modela crne kutije koji je fukusiran na to šta softver radi, model bele kutije je fokusiran na to kako softver radi.

Cilj testiranja ovim modelom jeste da se izvrše sve programske strukture i sve strukture podataka u softveru. Ovde se proverava samo kod, a ne specifikacija. Ovaj metod se može primeniti na svim nivoima testiranja, ali se najčešće primenjuje nakon metode crne kutije. Metoda bele kutije koriste i sami developeri, jer se očekuje da su implementirane komponente temeljno testirane pre nego se integrišu u softver, odnosno daju testerima na detaljno testiranje.

Pokrivenost testiranja (eng. coverage) je mera do koje je neki softver ili određena softverska komponenta istestirana nekim skupom testova, u smislu procenta obuhvaćenih stavki. U slučaju da pokrivenost nije 100%, potrebno je proširiti skup testova s novim testovima. Pokrivenost se povećava tako da se sistemski pišu novi testovi da bi se pokrili svi delovi softvera, odnosno da se svi delovi softvera izvrše bar jednom.

2.5. Nivoi testiranja softvera

Testiranje softvera, posebno velikih sistema, vrši se na više nivoa, od kojih svaki ima specifične ciljeve testiranja. Najčešći nivoi su: jedinično testiranje, integraciono testiranje, sistemsko testiranje i testiranje prihvatljivosti. Jedinično testiranje je testiranje funkcionalnosti programskih komponeti (jedinica) nezavisno od ostalih delova sistema. Jedinicama programa se smatraju funkcije ili klase. Programer koji piše kod obično i izvodi jedinično testiranje svog dela koda u razvojnom okruženju koristeći neki od okvira za jedinično testiranje.

Cilj ovakvog načina testiranja jeste da se detektuju funkcionalne i strukturne greške u odgovarajućoj jedinici, koje se odmah po nalaženju i poprave. Faza integracionog testiranja dolazi na kraju jediničnog testiranja. Integraciono testiranje je u praksi često zanemareno i ne obavi se na adekvatan način što dovodi do ozbiljnih problema. Integraciono testiranje je faza u kojoj se pojedinačni modeli i jedinice spajaju i testiraju zajedno kao celina.

Komponente koje ulaze u integraciono testiranje su već prošle jedinično testiranje. Ove komponente se grupišu i nad njima se izvršavaju posebno pripremljeni testovi. Ovo testiranje je važno iz razloga da se izbegnu svi potencijalni problemi koji mogu nastati prilikom spajanja komponenti u celinu.

Sistemsko testiranje je testiranje sistema kao celine, nakon što se uspešno završi kompletna integracija. Svrha sistemskog testiranja je verifikacija da softver kao celina ispunjava zahteve koji su definisani u specifikaciji zahteva sistema. Sistemsko testiranje obuhvata testove koji se definišu na osnovu specifikacije zahteva sistema i obično ga obavljaju najiskusniji testeri.

Ponekad se za ovaj zadatak angažuje nezavisan tim testera, da bi se obezbedila potpuna objektivnost pri sistemskom testiranju. Testiranje prihvatljivosti je faza koja se obavlja kada se softver pravi za određenu namenu i određenog klijenta.

Kod ovog nivoa testiranja sam klijent daje sud o tome da li su ispunjeni svi zahtevi i da li isporučeni softver obavlja traženu svrhu. Često uključuje alfa i beta testiranje. Alfa testiranje se vrši tokom samog razvoja programa kad se programerima odmah ukazuje na greške. Beta testiranje se vrši onda kada su sve moguće softverske greške nađene i popravljene i obavlja se u realnom okruženju u kome softver i treba da se koristi.

2.6. Automatsko i manuelno testiranje softvera

Manuelno testiranje podrazumeva ručno izvršavanje test skritpova različitim alatima u kojima se prate koraci testa i beleže test rezultati. Izvršavanje testa se smatra uspešnim ako je ponašanje sistema koji se testira u skladu sa očekivanim ponašanjem.

Nasuprot manuelnom je automatsko testiranje koje zahteva postojanje određenog koda koji je napisan da bi se automatizovali koraci pri izvršavanju određenog test skirpta. Manuelno i automatsko testiranje prate iste faze procesa testiranja i mogu se primeniti na različitim novima i tipovima testiranja.

2.7. Tipovi testiranja softvera

U tipove testiranja softvera spadaju: testiranje performansi sistema, test prihvatanja od strane korisnika (UAT), regresivno testiranje, statičko testiranje, smoke testing i sanity testiranje. Performanse obuhvataju vreme odaziva softvera, pouzdanost, upotreba resursa i skalabilnost. Testiranje performansi je tip testiranja koji verifikuje da se sistemski softver ponaša na odgovarajući način pod nekim očekivanim opterećenjem.

Cilj testiranja performansi nije pronalazak novih defekata, već da se eliminišu potencijalni problemi koji utiču na performanse sistema. Test prihvatanja od strane korisnika (eng. User acceptance testing ili Acceptance testing) je testiranje koje vrši klijent kada je sistem spreman za isporuku, nakon što se ispravila većina defekata pronađena u sistemskoj fazi testiranja.

Cilj ovog testiranja jeste da klijent stekne poverenje u sistem koji je implementiran. Proveravaju se funkcionalnosti prema specifikaciji zahteva i određuje se da li sistem ispunjava potrebe krajnjih korisnika. Regresivno testiranje je tip testiranja softvera čiji je cilj da potvrdi da nove promene u kodu softvera nisu uzrokovale greške u već postojećim funkcionalnostima sistema.

Ovaj tip testiranja podrazumeva ponavljanje testova koji su već izvršeni u prethodnim iteracijama, da bi bili sigurni da postojeće funkcionalnosti rade ispravno i nakon implementacije novih komponenti.

Statičko testiranje je skup tehnika kojim se poboljšava kvalitet softvera, kao i efikasnost i produktivnost samog procesa razvoja softvera. Ovaj tip testiranja se bazira na procesu statičkog pregleda (eng. review), s ciljem da se defekti pronađu što ranije u procesu razvoja softvera.

Smoke test je oblik testiranja softvera koji proverava da li osnovne funkcionalnosti softvera rade na unapred određen način. Ovaj oblik testiranja nije detaljan, niti ulazi u dubinu funkcionalnosti.

Izvršava se mali broj testova samo da bi se proverilo da li osnovne funkcije softvera rade. Ovaj oblik testiranja se koristi kao osnovna provera da li je softver dovoljno stabilan i spreman za dalje testiranje. Test zdravog razuma (eng. Sanity test) je sličan smoke testu. Ovi termini se često mešaju u testiranju softvera. Iako razlike između ova dva tipa nisu velike, one ipak postoje. Smoke test se koristi za testiranje build-a softvera da bi se verifikovao ispravan rad osnovnih funkcionalnosti i izvršava se pre bilo kojeg detaljnog testiranja, kako testeri ne bi gubili vreme u slučaju nekih kritičnih problema. Sanity test se izvršava nad relativno stabilnim softverom. Ovaj tip testiranja vrše testeri nakon primanja build-a u kome postoje izmene u kodu ili je dodata nova funkcionalnost.

3. JASMIN OKVIR

Jasmin (eng. Jasmine) je razvojni okvir vodjen ponašanjem koji omogućuje pisanje testova za testiranje koda napisanog u JavaScript programskom jeziku. Napravljen je tako da ne zavisi od drugih JavaScript razvojnih okvira i za njegovo korišćenje nije potreban objektni model dokumenta (eng. Document Object Model - DOM). Ima jednostavnu i veoma razumljivu sintaksu tako da je relativno jednostavan za korišćenje čak i za programere koji nemaju iskustva sa pisanjem testova.

Kako je ovo razvojni okvir namenjen za testiranje koda, Jasmin se može koristiti kao alat komandne linije koji automatski pronalazi napisane testove, pokreće ih i izveštava o rezultatima.

3.1. Razvoj softvera vođen ponašanjem

U softverskom inženjerstvu, razvoj zasnovan na ponašanju je agilni process razvoja softvera koji podstiče saradnju među programerima, testerima i kupcima u okviru softverskog projekta. On ohrabruje timove da koriste razgovor i konkretne primere za formalizovanje zajedničkog razumevanja o tome kako se aplikacija treba ponašati. Nastao je iz ravoja zasnovanog na testovima (eng. Test-driven devolopment). Razvoj vođen ponašanjem kombinuje opšte tehnike i principe razvoja zasnovanog na testovima sa idejama iz objektno-orijentisane analize i dizajna kako bi se omogućili programerima, testerima i ostalima koji učestvuju u razvoju softvera zajednički alati i zajednički proces za saradnju na razvoju softvera.

3.2. Jasmin funkcionalnosti

Postoji mnogo funkcionalnosti koje Jasmin donosi sa sobom, a u ovoj sekciji će biti nabrojane neki od njih. Osim ugrađenih funkcionalnosti, postoji i mogućnost implentiranja prilagođenih za potrebe određenog projekta. Možda i najjednostavnija metoda je toEqual. Ona, kao što i sam naziv kaže, proverava da li je ono što se poredi isto, ali ne nužno da li je i objekat isti.

Metoda toBe provera da li je ono što je poređeno pripada istom objektu, dok toEqual proverava samo običnu jednakost. Da bismo testirali da li je nešto tačno ili netačno, koristimo metode toBeTruthy i toBeFalsy. Treba napomenuti da je Jasminova evaluacija identična evaluaciji Javascript-a kada je reč o tačnosti ili netačnosti. Ovo znači da je true uvek tačno, kao i string „pas“ ili broj 9, ili bilo koji objekat.

Ukoliko želimo da negiramo neku metodu, poput metode toEqual ili metode toContain, možemo jednostavno iskoristiti prefiks .not. Ponekad je potrebno proveriti da li je element prisutan u okviru liste, ili u okviru string-a. Tada je moguće iskoristi metodu toContain. Pored već navedenih, postoji još mnoštvo metoda koje Jasmin donosi sa sobom, a

neke od njih su sledeće: toBeDefined i toBeUndefined (provera da li je ono što se proverava definisano ili nije), toBeNull (provera da li je prosleđena vrednost null), toBeNaN (provera da li prosleđena vrednost nije broj), toBeGreatherThan, toBeLessThan (komparacija prosleđenih vrednosti, koja ne radi nužno samo sa brojevima), toBeCloseTo (provera da li je broj dovoljno blizu drugom prosleđenom broju), toMatch(provera da li prosleđena vrednost odgovara određenom izrazu).

Postoje trenuci kada za testiranje nisu dovoljne već postojeće metoda Jasmin okvira. Tada se mogu kreirati prilagođene metode. Da bi se to uradilo, mora se na početku svakog testa dodati željena metoda. Još jedna od korisnih funkcionalnosti Jasmin okvira su metode BeforeEach i AfterEach. One omogućavaju da se izvrše određeni delovi koda, pre ili posle svakog testa. Ova funkcionalnost dosta pomaže da bi se postigao čistiji kod, kao i čišćenje varijabli nakon samog testa.

Kada kod postane dosta kompleksniji, samim tim postoji i mnogo više testova i oni su sve kompleksniji. Tada se mogu formirati grupe, podgrupe, podgrupe podgrupa, itd. Ovo je moguće izvesti tako što se describe blok, koji služi za opis testa, stavi unutar drugog describe okvira.

3.3. Jasmin okvir – testiranje Angular aplikacije

TestBed dolazi sa modulom koji je konfigurisan kao i svi ostali moduli u okviru Angular aplikacije. Moguće je deklarisati komponente, direktive, servise, kao i importovati druge module. Ono što je bitno kod TestBed-a, on dolazi sa metodom configureTestingModule koja prima definicije modula koji želimo da testiramo. Nakon što je konfigurisana komponenta koja će biti testirana, potrebno je renderovati komponentu, kako bi se moglo pristupati njenim delovima.

TestBed poseduje metodu createComponent, povratna vrednost ove metode je fixture, instanca klase ComponentFixture. Fixture sadrži komponentu u sebi i pruža interfejs i za instancu komponente i za renderovani DOM.

Moduli su centralni deo svake Angular aplikacije. Ipak, teški su za testiranje jer u njima neke tipične logike, već uglavnom samo konfiguracija. Moduli jesu klase, ali su one same po sebi najčeće prazne. Glavni deo modula se nalazi u delu označenim sa NgModule, jer se tu nalaze metapodaci (deklaracije, ulazne i izlazne komponente, provajderi, itd..). Ipak, u toku vremena, može doći do grešaka i u okviru modula, pa je potrebno napisati testove i za module. Ove greške mogu biti uhvaćene ranije pomenutim smoke testovima.

Testiranje običnih servisa je poprilično jednostavno, mogu se testirati njihove metode, ali ono što je veoma važno, potrebno je testirati servise koji komunicaraju sa backend-om aplikacije, i šalju HTTP zahteve. Prvi korak testa, nakon onih već poznatih, je da se pozove metoda nad servisom koja šalje HTTP zahtev, u ovom slučaju je to metoda getByCode koja služi da vrati valutu koja odgovara prosleđenom stringu od 3 karaktera.

U drugom koraku je korišćen HttpTestingController. Ovaj kontroler poseduje metode za nalaženje zahteva po različitim kriterijumima.

Najjednostavnija od njih je ona koja je korišćena, metoda expectOne, koja očekuje da nađe tačno jedan zahtev za zadati kriterijum.

Ova metoda vraća instancu klase TestRequest, a ukoliko ne pronađe nijedan odgovarajući zahtev, metoda expectOne baci grešku.

Treći korak je formiranje odgovora servera sa lažnim podacima. U ovom slučaju je to linija koda koja nad zahtevom poziva metodu flush. Uloga ove metoda jeste da simulira uspešni “200 OK“ odgovor servera.

Četvrti korak, pre same provere rezultata testa, je poziv metode verify nad kontrolerom. Uloga ove metode je provera da li je preostalo zahteva koji čekaju. U ovom slučaju je pronađen jedan zahtev metodom expectOne i na njega je odgovoreno metodom flush.

4. ANALIZA USPEŠNOSTI JASMIN ALATA

Jasmin alat je jedan od najkorišćenijih alata kada je reč o alatima kojima se testiraju web aplikacije, ali pored njega, još se koriste i sledeći alati: Jest, Mocha, Cypress, Puppeteer, AVA, StoryBook, Enzyme, itd.. Svi alati imaju svoje mane i prednosti. Ono što je potrebno uraditi je pažljivo odabrati alati koji bi trebalo koristiti. Najpopularniji alati za testiranje su Jest, Jasmin i Mocha. Jest okvir za testiranje jeste verovatno najbolji izbor kada se radi o web aplikacijama koje su pisane u React framework-u.

Razlog za to je prilično jednostavan, Jest u kombinaciji sa React-om donosi zgodnu osobinu, a to su regresivni vizuelni testovi kojima se lako otkrivaju slučajne greške nastale na korisničkom interfejsu. Jest ima opciju da beleži slike ekrana(eng. screenshots) i na taj način može da poredi jednu te istu komponentu tokom njene dorade. Takođe zahteva minimalnu konfiguraciju i veoma je dobro dokumentovan. Ono što je njegova najveća mana u odnosu na Jasmin, je ta što u poređenju sa Jasmin testnim alatom Jest ne podržava ni blizu toliko biblioteka i alata koje nekada mogu biti veoma korisne.

Popularnost Jest-a jeste propraćena povećanim korišćenjem React framework-a, ali se svakako mora reći da uz Jasmin testni okvir spada u najčeće korišćene i čini njegove korisnike zadovoljnim. Mocha je do pre nekoliko godina bila možda i najpopularniji alat za testiranje, ali se pojavom Jest-a, Jasmin i Cypress alata njena upotreba znatno smanjila. Ono što je najveća mana ovog alata je vreme koje potrebno za podešavanje i konfiguraciju. Takođe, generalne perfomanse nisu na najvišem nivou u poređenju sa ostalim alatima.

Cypress testni alat donosi sa sobom nekoliko zgodnih osobina u odnosu na druge testne alate, kao što su: mogućnost automatskog skrola, omogućava direktne promene DOM-a, čuva slike ekrana na svakom koraku, što omogućava programeru da proveri stanje na svakom koraku, itd..

Najveća mana ovog testnog alata jeste to što podržava samo JavaScript framework za pisanje testova i to što ne može da rukuje sa više otvorenih tabova u internet pregledaču, niti sa više otvorenih pregledača.

Jasmin okvir za testiranje ipak, uz Jest, donosi najviše a pritom ima najmanje mana. Kompatibilan je sa svakim framework-om, što ga čini najfleksibilnijim alatom. Podrška za Jasmin alat se može naći na mnogo mesta u formama biblioteka, blogova i video tutorijala. Jasmin nudi elegantna rešenja i paterne.

Dostupan je u svim internet pregledačima. Trenutno dva verovatno najpopularnija frontend framework-a su svakako React i Angular. Kada je reč o Angular-u, svakako najbolji izbor za testiranje jeste Jasmin okvir, dok je Jest ipak izbor kada se radi o React framework-u, zbog prethodno navedenih razloga.

5. ZAKLJUČAK

Testiranje je veoma važan deo razvoja jednog softvera. Iako svi teže ka tome da pišu kod bez grešaka, one su neizbežne. Pogotovo su problematične one greške koje se možda i ne primete pri samom razvoju softvera, već kada on završi kod klijenta. Takve greške se neretko jako teško otklanjaju i potrebni su sati ili dani da bi se otkrio uzrok istih. Takođe, napretkom tehnologije se aplikacije mnogo brže prave, kraći su rokovi za izradu, a to donesi i veću šansu za greške.

Ukoliko je reč o softveru koji je vezan za određene industrije, greške mogu imati fatalne posledice. Zbog svega navedenog testiranje je jako bitno. U ovom radu je korišćen Jasmine okvir za testiranje. Testiranje se izvršavalo nad Angular aplikacijom. Prikazani su primeri testova svih najvažnijih delova jedne Angular aplikacije kao što su komponente, servisi, pajpovi i moduli.

6. LITERATURA

[1] Evan Hahn, *JavaScript Testing with Jasmine*, 2013.

[2] Paulo Ragoha, *Jasmine JavaScript Testing*, 2018.

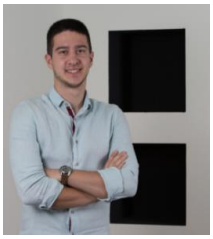
[3] Miodrag Živković, *Testiranje softvera*, 2018.

[4] https://www.innoq.com/en/blog/ts-jasmine-karma/

[5] https://testing-angular.com/

[6] https://medium.com/@turhan.oz/typescript-with-jasmine-easy-project-setup-530c7cc764e8

Kratka biografija:



Marko Vučković rođen je u Subotici 1995. godine. Završio je gimnaziju „Svetozar Marković“ u Subotici 2014. godine i upisao Fakultet Tehničkih Nauka u Novom Sadu iste godine. 2018 godine je završio osnovne akademske studije sa prosečnom ocenom 9.16.

kontakt:
markovuckovic1808@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/15BE08Djekic

RAZVOJ APLIKACIJE ZA DETEKTOVANjE SOFTVERSKIH OBRAZACA

DEVELOPMENT OF APPLICATION FOR DETECTION OF SOFTWARE PATTERNS

Nemanja Đekić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu demonstrirana je tehnika detektovanja softverskih obrazaca gde je implementirana aplikacija koja radi sa podacima koji predstavljaju „XMI“ reprezentaciju softverskog sistema. Aplikacija radi sa „NoSQL“ bazom podataka. Aplikacija je razvijena u „Microsoft Visual Studio“ programskom okruženju korišćenjem „C Sharp C#“ programskog jezika.*

Ključne reči: *Softverski obrasci, Algoritam, Detekcija*

Abstract – *This paper demonstrates the technique of software processing detection where an application is implemented that works with "XMI" representation of the software system. The application works with the "NoSQL" database. The application was developed in the "Microsoft Visual Studio" programming environment using the "C Sharp C #" programming language.*

Keywords: *Software Patterns, Algorithm, Detection*

1. UVOD

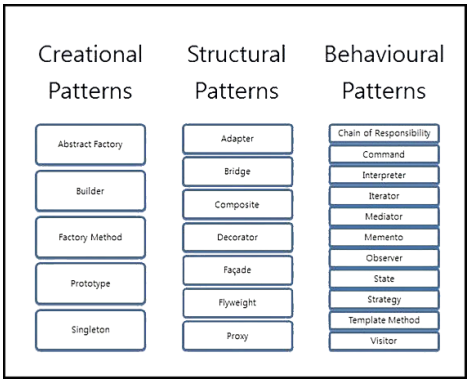
Objektno orijentisani obrasci predstavljaju dobro poznata rešenja za uobičajene probleme dizajna u datom kontekstu. Najpoznatija kolekcija objektno orijentisanih obrazaca sadržana je u knjizi „Gamma et al“.
Autori su prikupili i dokumentovali 23 uzorka obrazaca koje su predstavili kroz programske jezike „Smalltalk“ i „C++“ [1, 2]. Kroz ovaj rad biće sumirani određeni pristupi u analizi prisutnosti određenih objektno orijentisanih obrazaca u okviru raznih reprezentacija sistema ali će se analizirati pristup kao i aplikacija koja je nastala kao rezultat ovoga istraživanja.
Pristup u analizi zasniva se na višestruko redukciono strategiji baziranoj na podacima kao i meta podacima koji su nastali u procesu kreiranja klasnog dijagrama sistema.
Benefiti koji proizilaze iz upotrebe objektno orijentisanih dizajn obrazaca su višestruki pa tako iz ugla razumevanja aplikacija koje implementiraju određene obrasce možemo uočiti da obrasci pružaju informacije o rolama određenih klasa, razloge određenih veza među sastavnim delovima obrazca kao i preostalim delovima sistema.
Drugačije rečeno, otkrivanje obrazaca u okviru aplikacija predstavlja korak ka razumevanju procesa koji čine samu aplikaciju kao i dokumentaciju koja je opisuje.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Darko Čapko.

Shodno tome identifikacija obrazaca pruža uvid u strukturu softvera kao i uvid u mesta u softveru gde je moguće uvesti određene promene ali i proširenja. Štaviše sistem koji je dizajniran tako da koristi dobro poznate, dokumentovane i prihvaćene obrazce, pokazaće u produkciji dobra svojstva kao što su modularnost, razdvajanje odgovornosti, višestruke upotrebljivosti kao i lakoću proširenja. Pored ovoga upotreba određenih obrazaca može pomoći u davanju određenih informacija ali i naznaka o kvalitetu celokupnog sistema. Prisustvo dizajn obrazaca u rešenju treba da se reflektuje tako na sistem da izvlačenjem veza i uzorkovanjem dizajna aplikacije možemo da opravdamo izbor određenog rešenja u određenom sistemu te tako pojednostavimo izgradnju određenog konceptualnog modela sistema [3].

2. SOFTVERSKI OBRASCI

Trenutna upotreba izraza obrazac izvedena je iz pisanja arhitekte „Christopher Alexander“ koji je napisao i objavio više izdanja na temu dizajn obrazaca. Knjige su opisivale arhitekturu i urbanističko planiranje, ipak ideje su bile višestruko primenjive na više različitih disciplina pa tako i na razvoj softvera. Softverski obrasci postali su popularni i prihvatljivi sa pojavom knjige „Gamma et al“ 1995. godine, te od tada interesovanje oko obrazaca, jezika i načina na koji se obrasci implementiraju raste eksponencijalno i danas je prisutno gotovo u svakom softveru [2].
Prema klasifijaciji predloženoj u knjizi „Gamma“, obrasci mogu se klasifikovati na kreacione, strukturalne i obrasce ponašanja. Kreacioni obrasci obuhvataju stvaranja objekata, strukturalni obrasci obuhvataju klase i objekte, odnosno njihov sastav, dok se obrasci ponašanja bave načinima na koje klase raspoređuju odgovornost međusobno te proizvode rezultat [4]. Na slici 2.1 možemo videti klasifikaciju softverskih obrazaca koji se aktivno koriste u industriji.



Slika 2.1 *Objektno orijentisani obrasci [9]*

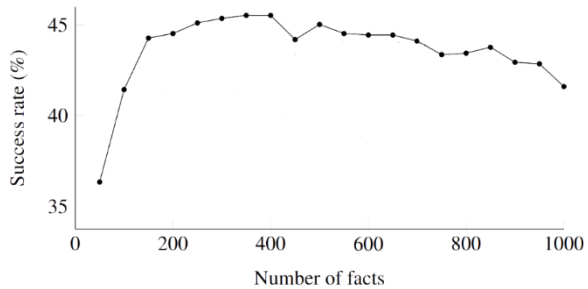
3. ALATI ZA DETEKTOVANJE OBRASACA

Gledano kroz istoriju softvera, od 1990. godine pa do danas razvila su se ali i dalje se razvijaju rešenja za detektovanje softverskih obrazaca u sistemima kako bi se olakšao pristup, promena ali i unapređenje rešenja problema određenog sistema.

U ovome radu biće opisano nekoliko rešenja koja postoje u industriji, od alata koji analiziraju „UML“ dijagrame zatim alata koji analiziraju data ograničenja sistema te na osnovu toga detektuju obrasce zatim rešenja koja pomoću meta podataka sistema daju aproksimaciju prisustva obrazaca u sistemu, do rešenja koja koriste mašinsko učenje kako bi detektovala prisustvo obrazaca u sistemu. U daljem tekstu biće dat kratak opis jednog od rešenja.

3.1 LAMBDES

“LAMBDES” (eng. Logic Analyser of Models Based on Descriptive Semantic) je prototip softverskog alata koji radi sa „UML“ modelima tako što ih prevodi u „FOL“ (eng. First order logic) sisteme te tako radi analizu i detekciju obrazaca određenim algoritmom te dobijene rezultate upoređuje sa već poznatim i fiksiranim parametrima alata koji opisuju obrasce te dalje sve dobijene podatke prosleđuje u „SPASS“ (eng. Synergetic Prover Augmenting Superposition with Sorts) koji je zapravo automatizovan sistem dokazivanja teorema te on dalje proizvodi krajnji rezultat analize.
Alat je skalabilan i veoma se lako integriše u ostale alate slične namene. Alat funkcioniše tako što radi prevođenje dijagrama, zatim određenim algoritmom pravi određene skupove pravila koji predstavljaju semantiku sistema.
Na osnovu statičke analize celokupnog sistema alat generiše skup pomoćnih konstanti koji mu pomaže u analizi. Korisnik alata je takođe u mogućnosti da navede određena ograničenja kako bi alat proizveo što bolje rešenje.
Kada su prepoznata sva pravila, konstante i ograničenja alat radi analizu i daje određene rezultate. Ukoliko korisnik alata nije zadovoljan rezultatima analize on je u mogućnosti da postavi nova ograničenja koja bi dovela do boljih rezultata analize.
Ovakav pristup detektovanju obrazaca prvi put je viđen upravo u ovome alatu, gde zapravo sam korisnik ima veoma veliku ulogu u samom radu i direktno utiče na detekciju obrazaca [5].
Na slici 3.1 prikazana je stopa uspešnosti alata u detekciji softverskih obrazaca u odnosu na broj raspoloživih podataka na osnovu kojih se vrši detekcija.

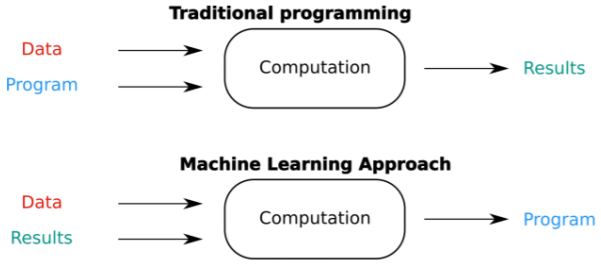


Slika 3.1 *Uspešnost „SPASS” sistema [5]*

4. MAŠINSKO UČENJE KAO ALAT ZA DETEKCIJU OBRAZACA

Obrasci softverskog dizajna su apstraktni opisi rešenja problema koji se ponavljaju u softverskoj industriji. Informacije o tome gde se koji obrazac primenjuje veoma je su korisne i važne za softversko održavanje i evoluciju. Ove informacije se obično dobijaju korišćenjem određenih alata koji su skalabilni i daju poprilično dobre rezultate ali i dalje zahtevaju da sam korisnik alata definiše određene parametre u radu. Ovo znači da postojeća rešenja za detekciju obrazaca nisu gotovo ni malo automatizovani procesi te sve informacije koje izgenerišu često budu samo jednom upotrebljene ali često i izgubljene. Upravo zbog toga, mašinsko učenje kao alat donosi revoluciju u svet detektovanja obrazaca u softverskim sistemima. Mašinsko učenje oslanja se na već postojeće podatke o softverskim obrascima, alate koji već postoje u industriji za detekciju softverskih obrazaca ali naravno i veštačke neuronske mreže koje su obučene da dobijene podatke analiziraju.

Predloženi pristup prepoznavanja dizajn obrazaca na osnovu metoda mašinskog učenja započinje prvom fazom u kojoj se smanjuje prostor za pretragu identifikovanjem skupa klasa kandidata za svaku ulogu u svakom dizajn obrascu. Zatim se u drugoj fazi za sve moguće kombinacije odnosno kandidate dizajn obrazaca proverava da li su oni zapravo valjana instanca nekog od dizajn obrazaca. Odvojena veštačka neuralna mreža „ANN“ (eng. Artificial neural network) se zatim obučava paralelno sa različitim vektorima obeležja koji dekorišu valjane instance dizajn obrazaca, pored ovoga neuralna mreža radi i sa podacima koji su nastali kao rezultat već postojećih rešenja za detektovanje obrazaca u softverskim sistemima. Kada se obrada podataka izvrši, ukoliko su neuronske mreže u procesu paralelnog obučavanja imale stopu uspešnosti veću od pedeset procenata, možemo očekivati da će alat detetektovati barem neki softverski obrazac sa velikom sigurnošću [2, 3]. Na slici 4.1 ilustrovana je razlika u pristupu između tradicionalnog programiranja i pristupa programiranju koji nudi mašinsko učenje.



Slika 4.1 *Pristup tradicionalnog programiranja i mašinskog učenja [5]*

Ono što je bila motivacija za izradu prvih alata koji su koristili neuronske mreže jesu zapravo dva povezana motiva i cilja ka kojima se težilo. Prvi je da se u potpunosti iskoristi snaga i sposobnost metoda mašinskog učenja i upotrebe neuronske mreže u rešavanju problema odnosno prepoznavanju softverskih obrazaca. Uprkos veoma poznatom i velikom uspehu neuronskih mreža u

rešavanju drugih problema, problem prepoznavanja dizajn obrazaca bio je potpuno nov u svetu neuronskih mreža. Ono za šta su se najviše do tog momenta koristile neuronske mreže jeste zapravo filtriranje lažnih pozitivnih rezultata ili smanjivanje prostora za pretragu kako bi doveli do pronalaska rešenja problema stavljenog pred njih. Drugi motiv razvoja ovakvog pristupa jeste da se u potpunosti nauče pravila i funkcije prepoznavanja instanci softverskih obrazaca u sistemima koji se već izvršavaju na nekoj mašini [3,7].

Gotovo sve tehnike koje su do tada postojale za detektovanje softverskih obrazaca zavisile su u različitoj meri od pravila i osobina izvedenih iz teorijskog opisa softverskih obrazaca što ne znači da su eksplicitno te osobine kroz implementaciju dospele u softver. Neke od tih tehnika zasnivali su se na tome da su teorijski opisi obrazaca u potpunosti prisutni u već implementiranim softverima, što naravno nije slučaj [6].

Upravo zbog ovakvih pristupa koji su kroz praksu pokazali veoma malu stopu uspeha, mašinsko učenje i neuronske mreže krenule su drugačijom putanjom. One su krenule od analize ne teorije dizajn obrazaca nego samih instanici dizajn obrazaca u okviru softvera. Ovakav pristup dao je neočekivano dobre rezultate i gotovo neuporedivo veći set podataka koji podrazumeva obeležja i pravila nad kojima su neuronske mreže mogle da rade i da se treniraju. Treniranje neuronskih mreža ovakvim pristupom pomoglo je izbegavanju problema lažnih pozitivnih poklapanja ali je donelo i podršku za veću granularnost rešenja, što postojeća rešenja gotovo nisu ni imala u vreme kada je prvi alat ovakve prirode nastajao [3,8].

5. APLIKACIJA

U okviru master rada pravljena je aplikacija radi demonstracije tehnike detektovanja softverskih obrazaca. Aplikacija radi nad podacima dobijenim modelovanjem softvera korišćenjem alata „Enterprise Architect“. Aplikacija se sastoji iz dva osnovna dela. Prvi deo aplikacije zadužen je da modeluje podatke koji se mogu naći u „XMI“ reprezentaciji sistema koji alat generiše na osnovu modela sistema odnosno dijagrama klasa sistema, dok je drugi deo aplikacije zadužen za parsiranje podataka dobijenih kroz reprezentaciju sistema, detektovanje obrazaca u okviru tih podataka na osnovu određenog algoritma kao i za generisanje koda na osnovu dobijenih podataka.

Algoritam se sastoji iz dva dela:

1. Transformacija „XMI“ reprezentacije softvera u objektnu reprezentaciju pogodnu za dalju analizu.
2. Analiza dobijenih podataka i prepoznavanje softverskih obrazaca.

Aplikacija se nalazi u okviru jednog rešenja i sastoji se iz tri zasebna dela. Prvi deo rešenja sadrži sve klase koje su zadužene za smeštanje informacija dobijenih parsiranjem „XMI“ reprezentacije softvera. Pored smeštanja informacija, pojedine klase zadužene su i za dalju separaciju izvučenih podataka. Zadatak drugog dela jeste da detaljno analizira „XMI“ reprezentaciju softvera odnosno dijagrama klasa, da detektuje prisustvo softverskih obrazaca, kao i da generiše kod na osnovu dobijenih podataka. Algoritam radi tako što prolazi kroz sve prikupljene podatke, i na osnovu informacija o imenima klasa, imenima metoda, imenima parametara metoda, imenima polja u okviru klasa nasleđivanja u okviru klasa, povratnih vrednosti, pristupnosti polja, pristupnosti klasa, pristupnosti interfejsa, vrši analizu i uz određenu logiku detektuje prisistvo softverski obrazaca. Ukoliko je aplikacija detektovala prisustvo softverskog obrasca, rezultat upisuje u bazu podataka i nastavlja dalju analizu podataka.

Kako sam algoritam prati svaku vezu između klasa, metoda, povratnih vrednosti, vrednosti parametara, preciznost algoritma nije potpuna te je pojavljivanje lažnih pozitivnih rezultata moguće. Ono što je bitno da se napomene u vezi algoritma za detektovanje obrazaca koji je implementiran u okviru ovoga rešenja jeste to da za svaki obrazac algoritam ima kolekcije potencijalnih imena polja, metoda, getera i setera koje se koriste u softverskoj industriji. Ove kolekcije nisu od primarnog značaja za rad algoritma i veoma malo utiču na ishod, ali su upotrebljene kako bi se smanjila stopa detektovanja lažnih pozitivnih rezultata. Treći deo aplikacije jeste sam korisnički interfejs koji je implementiran tako da izvrši sve potrebne validacije pre nego to sam algoritam detekcije i generisanja koda izvrši.

U daljem tekstu biće dat opis kao i pseudokod detekcije jednog od podržanih obrazaca. Obrazac uzorak, softver detektuje tako što prolazi kroz sva polja u okviru svih klasa i mapira ih. Uz mapiranje samih polja algoritam za svako polje vezuje i pravo pristupa tom polju kao i naziv. Posle mapiranja polja algoritam ponovo prolazi kroz sve podatke i mapira sve getere i setere u okviru klasa. Ukoliko se utvrdi da postoji veza između polja koje je privatno u okviru neke klase i getera u okviru te klase koji ima povratnu vrednost istu kao što je i tip polja, klasa se obeležava kao kandidat za softverski obrazac uzorak. Ukoliko pored toga sam naziv getera nosi neki od naziva koji se obično koriste kada se implementira obrazac uzorak, klasa se obeležava kao kandidat za obrazac uzorak. Procedura detektovanja može se predstaviti i pomoću pseudokoda:

```
foreach (x in fields)
if (x.Type == y || posibleNames.contains(y.Name))
        Mark as possible candidate for pattern.

foreach (x in properties)
if (x.Type == this.Type)
        Mark as possible candidate for pattern.

foreach (x in markedEntities)
if (x.MarkForReason == reasonOfDetection)
        Add to collection of detections.

Check collection for detection count and proclaim
detection if possible.
```

5.1 Testiranje softvera

U daljem tekstu biće opisana metodologija testiranja softvera koji je proizišao iz ovoga rada kao i stopa

uspešnosti koju je softver imao u detekciji softverskih obrazaca.

Testiranje softvera je izvršeno kroz par koraka a oni su:

1. Pronalaženje implementacija softverskih obrazaca u „C#“ programskom jeziku.

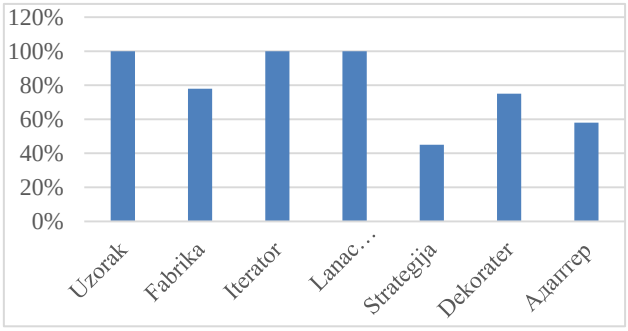
2. Prebacivanje implementacija u dijagrame klasa na osnovu kojih se radi analiza prisustva softverskih obrazaca.

3. Puštanje softvera u rad i analiza dobijenih rezultata.

U prvom koraku za svaki softverski obrazac, prikupljeno je najmanje četiri različite implementacije softverskog obrasca. Implementacije su prikupljene iz različitih izvora kako bi se eliminisala mogućnost ponavljanja. Nakon prikupljanja, svaka implementacija posebno je prebačena u dijagram klasa pomoću softvera „Enterprise Arhitect“, te je nakon toga na osnovu dijagrama klasa generisana „XMI“ reprezentacija dijagrama.

Nakon generisanja svaka reprezentacija posebno je upotrebljena kao izvor podataka softvera za detektovanje softverskih obrazaca. U tabeli 1 prikazani su rezultati testiranja. Ono što može da se zaključi jeste da softver ima veoma veliku stopu uspešnosti u detekciji softverskih obrazaca čija implementacija nije direktno zavisna od implementacije tela metoda, već se detekcija može odraditi na osnovu tipova povratnih vrednosti, parametara metoda i polja ali i na osnovu imena koja su upotrebljena u implementaciji kao i ostvarenim vezama između samih klasa, kao što je to slučaj za obrazac uzorak.

Tabela 1: *Rezultati testiranja*



6. ZAKLJUČAK

Sa pojavom objektno orijentisanog pristupa u softveru, softverski obrasci napravili su pravu pometnju u razvoju softvera. Klasične strukture koje su podrazumevale jedan ogroman fajl sa puno odgovornosti gotovo su nestale. Novi pristup koji su doneli softverski obrasci razdvojio je odgovornosti, povećao je skalabilnost, doneo je bolje performanse.

Dalji razvoj na ovome polju gotovo je neminovan i sa pojavom novih tehnologija prisustvo softverskih obrazaca je zagarantovano.

Za potrebe ovog rada napravljena je implementacija jedne od tehnika detekcije softverskih obrazaca.

Cilj implementacije bila je demonstracija rada tehnike u industriji kroz manji primer, nad relativno manjem obimu podataka. Dalji pravci u istraživanju u okviru ovoga rada jesu dublje razumevanje integracije mašinskog učenja sa detekcijom softverskih obrazaca kao i softverski ali i hardverski zahtevi koje ta integracija donosi sa sobom.

7. LITERATURA

[1] G Guéhéneuc, Y. G., & Antoniol, G. Demima, A multilayered approach for design pattern identification, *IEEE transactions on software engineering* 34.5: 667-68, 2008.

[2] Kramer, C., & Prechelt, L., Design recovery by automated search for structural design patterns in object-oriented software, *Proceedings of WCRE'96: 4rd Working Conference on Reverse Engineering*, 1996.

[3] Fabry, J., & Mens, T., Language-independent detection of object-oriented design patterns, 2004.

[4] Srinivasan, S. Design patterns in object-oriented frameworks. Computer, 1999.

[5] Zhu, H., Bayley, I., Shan, L., & Amphlett, R., Tool support for design pattern recognition at model level, *2009 33rd Annual IEEE International Computer Software and Applications Conference*, 2009.

[6] Jing Dong, J., Sun, Y., & Zhao, Y., Design pattern detection by template matching, *Proceedings of the 2008 ACM symposium on Applied computing*, 2008.

[7] Afloz Chakure, Introduction to Machine Learning, 2019

[8] Prakash, N., Manconi, A., & Loew, S., Mapping landslides on EO data: Performance of deep learning models vs. traditional machine learning models, *Remote Sensing* 12.3: 346, 2020.

[9] Fowler, M., Patterns [software patterns], 2003

Kratka biografija:

Nemanja Đekić rođen je u Subotici 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Razvoj aplikacije za detektovanje softverskih obrazaca odbranio je 2021.god.

kontakt:
nemanja.djekic.1995.2019@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/15BE09Ercegovac

TESTIRANJE JEZIKA SPECIFIČNOG ZA DOMEN ENERGETSKE RAZMENE BEZ POSREDNIKA BAZIRANOG NA JETBRAINS MPS MBDDR I KERNELF SPECIFIKACIJI

TESTING OF A DOMAIN SPECIFIC LANGUAGE FOR P2P ENERGY TRADING BASED ON JETBRAINS MPS MBDDR AND KERNELF SPECIFICATIONS

Marko Ercegovac, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu će biti predstavljene mogućnosti primene KernelF jezika na pravljenje i testiranje DSL-a za domen P2P energy trading-a.*

Abstract – *The paper presents application of the KernelF language to the creation and testing of DSL (Domain-specific language) for the domain of P2P energy trading.*

Ključne reči: *domenski jezici, blokčejn, energetska razmena, pametni ugovori, KernelF, MPS.*

Keywords: *domain specific lanugage, blockchain, energy P2P trading, KernelF, MPS.*

1. UVOD

U ovom radu biće predstavljeno testiranje jezika specifičnog za domen energetske razmene bez posrednika implementacijom pametnih ugovora (Smart Contract-a) i korišćenjem interpretera uz podršku MPS (MethaProgramingSystem) okruženja mbeddr platforme i KernelF *embeddable* jezika. U daljem izlaganju, umesto pojma „energetska razmena bez korisnika" biće korišćen izraz "*P2P Energy trading* ".

Smart contract ili pametni ugovor je računarski program ili transakcioni protokol koji se izvršava automatski u skladu sa pravilima opisanim u sadržaju ugovora [1]. Pametni ugovori se izvršavaju na *block chain*-u koji predstavlja distribuiranu bazu podataka koja je transparentna svim korisnicima sistema zahvaljujući decentralizaciji sistema i kriptografiji [2]. Jedan od najpoznatijih *blockchain*-ova je Etherium *blockchain* koji predstavlja decentralizovanu mrežu koja nije pod kontrolom nijedne centralne organizacije i pomoću nje i jezika Solidity koji komunicira direktno sa Etherium *blockchain*-om moguće je pravljenje decentralizovanih aplikacija. Pisanje dobrih pametnih ugovora predstavlja izazov i programeri moraju detaljno testirati pametne ugovore pre njihove distribucije. Zbog sve većeg razvoja decentralizovanih aplikacija, neophodno je omogućiti automatizaciju razvoja i pisanja pametnih ugovora i povezati domenskog eksperta i programera.

U ovom radu će biti ukratko predstavljen domenski jezik sa interpreterom za opisivanje pametnih ugovora i automatski testovi koji su razvijeni za podršku daljeg razvoja i testiranja jezika. DSL (*Domain specific language* – domenski jezik) je programski jezik koji nudi softverska rešenja u nekom određenom domenu. Domenskim jezicima se mogu smatrati i saobraćajni znakovi ili notni sistem u muzici. Domenski jezici postaju sve popularniji zbog direktne povezanosti programera i domenskog eksperta. Domenskim ekspertima je omogućeno da povećaju svoju produktivnost i do 10 puta.

KernelF je funkcionalan jezik izgrađen na osnvu MPS-a. Dizajniran je da bude proširiv i ugradiv kao podrška jezgru domenskog jezika. KernelF je korišćen u širokom spektru jezika uključujući domenske jezike u oblasti medicine, finansija, obračuna plata, pametnih ugovora [3]. Jedni od najpoznatijih projekata izgrađenih na jezgru KernelF-a su Voluntis Healtcare i DATEV Payroll aplikacije. Voluntis je francusko-američka firma koja je napravila Healtcare platformu za lečenje personalizovanom digitalnom terapijom [4]. Koristeći algoritme, kako bi pronašli pravu dozu za pacijenta, platforma koristi već postojeće medicinske algoritme koji su se dobro pokazali u praksi. Aplikacija funkcioniše tako što telefon prikuplja sve podatke vezane za pacijenta, koje analizira i na osnovu medicinskih algoritama formira terapiju koju može da vidi lekarski tim i u svakom trenutku da kontaktira korisnika aplikacije. Projekat je nastao kao *startup* izgrađen na jezgru KernelF jezika. DATEV je registrovano društvo koje je pre svega pružalac tehničkih informacionih usluga za poreze, računovodstvo i advokate. U početku je bio dobavljač usluga, a sada softver direktno pruža usluge krajnjim korisnicima. Politika DATEV-a se zasniva u pružanju usluga na poreskom tržištu.

2. ENERGETSKI P2P TRADING

P2P (*Peer to Peer Service*) predstavlja decentralizovanu platformu gde dva pojedinca uzajamno interaguju bez prisustva trećeg lica. Umesto toga prodavac i kupac vrše transakcije direktno jedan ka drugome preko P2P servisa [5].

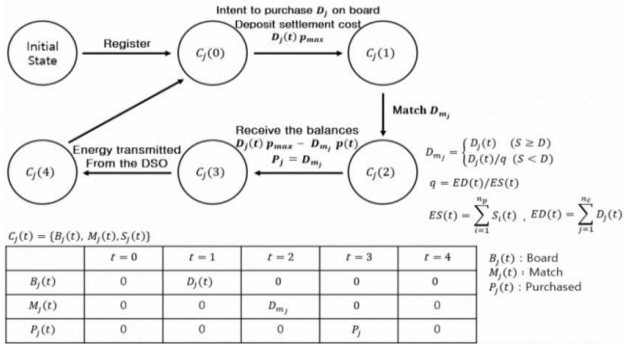
Osnovni koncepti energetske P2P razmene su dinamičko izračunavanje cene, DSO (*distribution system operator*) i korisnici koji učestvuju u razmeni energije (*Consumers*, *Prosumers*). Sa povećanim razvojem čiste (zelene) energije tradicionalni potrošači (*Consumers*) postaju proizvođači-potrošači (*Prosumers*) koji, koristeći foto-

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

ćelije ili energiju vetra, mogu da generišu energiju, skladište je i prodaju višak energije koju nisu potrošili. Veliki problem predstavljaju zakoni koji još uvek nisu usklađeni u odnosu na visok razvoj i potražnju za zelenom energijom. Sam proces razmene i prodaje energije ne može da bude siguran i pouzdan bez nekog operatera od poverenja kao što je DSO [6]. Svaki od učesnika u energetskoj razmeni mora proći kroz nekoliko stanja. To stanje isključivo zavisi da li je korisnik *Consumer* (slika 1) ili *Prosumer* (slika 2).

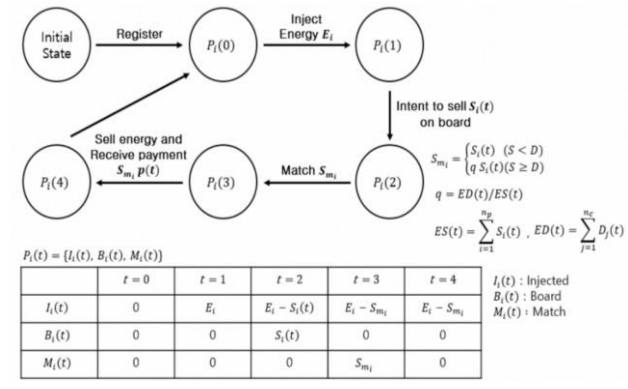
Na samom početku energetske razmene korisnik se registruje u sistem. Nakon toga prelazi u stanje *injected* (u tom stanju korisnik odlučuje koliko energije želi da proda), a zatim prelazi u stanje *on board* u koje objavljuje prodaju.

Ukoliko se nađe potencijalni *Consumer* koji kupuje tu količinu energije, prelazi u stanje *match* (upareno stanje) nakon kojeg prodaje energiju i dobija isplatu u odnosu na količinu prodate energije. Nakon uspešno obavljene energetske razmene proces se završava.



	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$
$B_j(t)$	0	$D_j(t)$	0	0	0
$M_j(t)$	0	0	D_{m_j}	0	0
$P_j(t)$	0	0	0	P_j	0

Slika 1. *Prikaz dijagrama stanja consumer-a u toku energetske razmene [6]*



	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$
$I_i(t)$	0	E_i	$E_i - S_i(t)$	$E_i - S_{m_i}$	$E_i - S_{m_i}$
$B_i(t)$	0	0	$S_i(t)$	0	0
$M_i(t)$	0	0	0	S_{m_i}	0

Slika 2. *Prikaz dijagrama stanja prosumer-a u toku energetske razmene [6]*

Bilo bi nezgodno u svakom mometu i za svaku pojedinačnu razmenu da potrošači i potrošači-proizvođači ponude ili traže za svaki period određenu vrednost. Da bi ovo izbegli, u sistemu se određuje pojedinačna cena [6]. Pojedinačna cena se određuje kao funkcija ukupne potražnje i ukupne ponude i jedna takva cena se koristi u toku procesa razmene.

Na samom početku procesa razmene označićemo totalnu ponudu (*supply*) sa ES(t) i totalnu potražnju (*demand*) sa ED(t) kao što je predstavljeno u jednačinama (1) i (2) [6].

$$ES(t) = \sum_{i=1}^{n_p} S_i(t)\ (S_i \geq 0) \quad (1)$$

$$ED(t) = \sum_{j=1}^{n_c} D_j(t)\ \left(D_j \geq 0\right) \quad (2)$$

$S_i(t)$ totalna ponuda od *prosumer*-a, $D_j(t)$ totalna potražnja od *consumer*-a i n_p broj *prosumer*-a i n_c broj *consumer*-a [6]. U formulama (3) i (4) su prikazani odnos R(t) i razlika D(t) između totalne potražnje (*supply*) i totalne ponude (*demand*) [6].

$$R(t) = \frac{ED(t)}{ES(t)} \quad (3)$$

$$D(t) = ED(t) - ES(t) \quad (4)$$

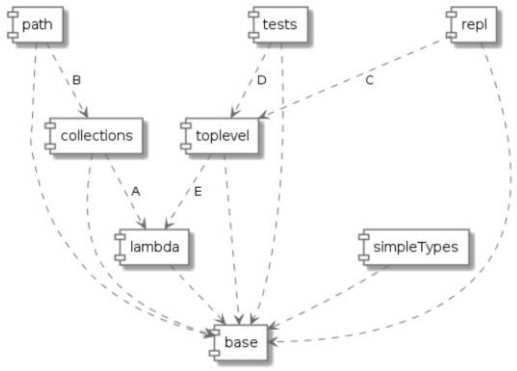
Chekired, Khoukhi and Mouftah [7] su predložili dinamičko izračunavanje cene korišćenjem R(t) i D(t) što je iskorišćeno u radu i na osnovu kog je izneta fukcija za dinamičko izračunavanje cene (5).

$$p(t) = \frac{2}{\pi}(p_{con}) \cdot tan^{-1}((lnR(t))^k) \ + p_{balance} \quad (5)$$

$p_{balance}$ je cena kada je totalna potražnja jednaka totalnoj ponudi to jest kada je R(t) = 1. p_{con} određuje opseg cene, dok koeficijent k „kontroliše“ opseg cene [6]. Formula postiže stabilnu cenu energije koja se ne menja puno u odnosu na ponudu i potražnju.

3. OSNOVNI KONCEPTI JEZIKA

KernelF je nastao kao potreba da se implementira jezik u MPS-u koji ima predefinisane principe: tipove podataka, promenljive, funkcije itd. Sam jezik KernelF je izgrađen od nekoliko manjih jezika koji su predstavljeni konceptima, što se može videti na slici 3.



Slika 3. *Kolekcija jezika u MPS-u od kojih se KernelF sastoji [8]*

KernelF je moguće proširiti drugim jezicima u MPS-u, takođe moguće je izgraditi DSL samo na osnovu KernelF-a. Jezik je izvršiv u javi i to preko javine virtualne mašine. KernelF predstavlja grupu različitih jezika implementiranih u MPS-u. Osnovni koncepti KernelF-a korišćeni u ovom radu su: *simple types*, *enumerations*, *state machines*, *functions*, *boxes*, *transactions* i *unit tests*. Postoje tri osnovna tipa koja koristi KernelF, a to su: *boolean*, *number* i *string*. *String* tip podržava interpolaciju

(konkatenaciju) *string*-ova, dok je *number* poseban tip koji sadrži opseg i preciznost. Ukoliko broj nije u dobrom opsegu program prijavljuje greške. Osnovne operacije u KernelF-u su unarne i binarne i koristi se infiksna notacija. KernelF podržava kolekcije i to liste, setove i mape. Tip tag je posebna vrsta tipa u KernelF-u na osnovu kog se može implementirati *custom* tip neke promenljive. Enumeracije koje podržava KernelF su: regularne i *valued flavors*. Enumeracije su potrebne za opisivanje stanja (*register*, *injected*, *on board*, *matched*, *purchased*).

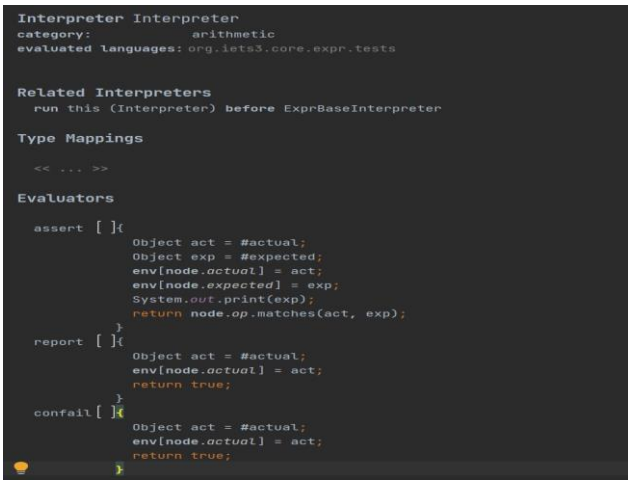
Konačni automati (*State machines*) su najbitniji koncept korišćen u ovom radu. Da bi se opisala pomenuta stanja *Prosumer*-a i *Consumer*-a koriste se konačni automati koje predstavljaju promenljivu (*mutable*) strukturu podataka. Konačni automati sadrže stanja, događaje i varijable i definisani su za jednu virtualnu mašinu. Događaje je moguće pozivati samo iz stanja u kome su ti događaji definisani. Svaki konačni automat ima inicijalno stanje sa inicijalnim vrednostima.

Funkcije u KernelF-u sadrže ime, listu argumenata, opcionu povratnu vrednost i telo funkcije. Takođe, funkcije pamte efekte i KernelF uvodi koncepte R-*read*, RM-*read/modify* koji omogućavaju na neki način kontrolu pristupa nad funkcijama. U koncepte KernelF-a uveden je tip nepromenljivog (*immutable*) podatka kao što je *boxes* koji nakon kreiranja nije moguće izmeniti. Korišćen je i koncept transakcija koje iplementira KernelF. Transakcioni blok je isti kao i obični blok ali ako nešto „pukne“ prilikom izvršavanja, sve izmene nad podacima se ponište unutar bloka. Testiranje je izuzetno bitno prilikom razvoja DSL-a. Testovi pomažu da se napiše dobar kod i da se proveri funkcionisanje koda. Testovi se sastoje od imena testa i različitih *elemenata* kao što su: *assertion, confail, report*. Klasično poređenje se postiže sa elementom *assert* koji poredi vrednost koju očekujemo sa dobijenom vrednošću. *Confail* proverava da li je test pukao (*fail*). *Report* vraća vrednost i tip elementa koji je testiran.

4. INSTALACIJA I KONFIGURACIJA

U ovom radu opisana je detaljna instalacija MPS-a sa sajta https://www.jetbrains.com/mps/ kao i upustvo kako instalirati mbeddr i IETS3 sa https://build.mbeddr.com. Takođe je opisan proces instalacije i konfiguracije KernelF-a. IETS3 predstavlja trenutnu verziju KernelF-a i stalno je u daljem razvoju i na sajtu https://build.mbeddr.com moguće je videti razne repozitorijume i različite artefakte. Da bi se implemetirao mbbeddr neophodno je sa glavnog repozitorijuma skinuti sa *master* grane na putanji mbbeddr/Main/Platform/master *zip* folder. Za IETS3 neophodno je skinuti sa repozitorijuma IETS3 Open Source/Build IETS3 opensource/master *zip* folder.
Preporučljivo je koristiti najnoviju verziju MPS-a, kao i najnovije verzije sa pomenutih repozitorija. Nakon uspešene instalacije *plugins*-a MPS JetBrains će prikazati i učitati prilikom pokretanja pomenute *plugin*-e.

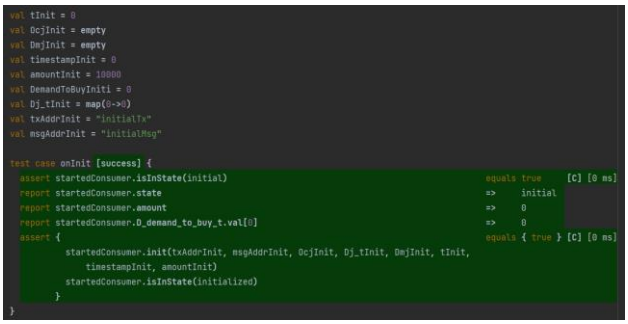
Nakon pokretanja MPS-a pojavi se prozor u kome se bira da li se pravi *solution* ili *language* projekat. Pošto je KernelF izgrađen kao predefinisani language projekat, potrebno je izabrati *solution* (rešenje) i nakon toga importovati pomenute koncepte. Moguće je napraviti i prazan projekat (*empty*), a naknadno da se izabere *solution* projekat. Da bi se moglo uopšte pisati u KernelF-u koriste se *Library* fajlovi koji se pozivaju tako što se implementira predefinisani koncept org.iets3.core.expr.toplevel, koje je za implementaciju testova neophodno importovati. Da bi testovi uopšte funkcionisali neophodno je implementirati interpreter testova (slika 4) koji je u ovom radu implementiran i opisana je detaljno njegova konfiguracija. Interpreter testova se pokreće pomoću Ctrl + Alt + Enter.



Slika 4. *Prikaz implementacije interpretera za testove u KernelF-u*

5. IMPLEMENTACIJA REŠENJA

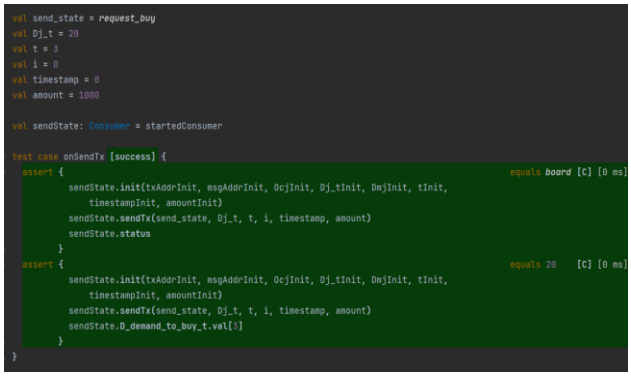
Celokupna implementacija jezika je prikazana u delovima i objašnjenjima šta koji deo koda radi [8] . Celokupan kod rešenja nije prikazan ali je prikazano njegovo uspešno izvršavanje u sklopu *unit* testova. Rešenje se sastoji iz nekoliko *Library* fajlova koji su međusobno povezani i moguće je koristiti definisane koncepte iz različitih *Library* fajlova. Opisani su u potpunosti Prosumeri i Consumeri sa njihovim stanjima kao i događaji koji se izvršavaju kad se pomenuti korisnici nalaze u različitim stanjima.



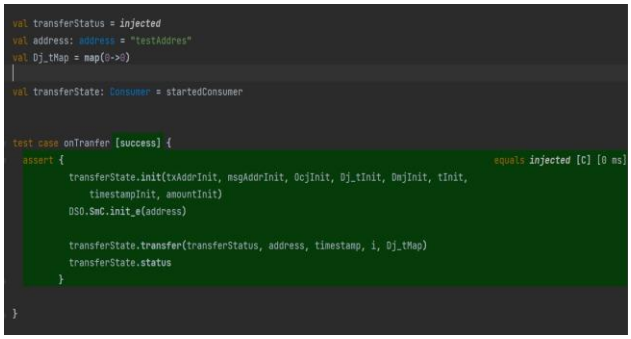
Slika br.5 *Test Consumer-a na inicijalni događaj*

Implementirani su događaji za prelazak stanja, kao i događaji za međusobnu komunikaciju i razmenu energije. Implemetirana je funkcionalnost registracije novih korisnika u sistem. Celokupno rešenje je prikazano kao DSL koji olakšava korišćenje nekih od već pomenutih jezika za pisanje pametnih ugovora (*Smart Contract*-a). Jezik je jednostavan, mali i modularno implemetiran. Svi testovi su uspešno izvršeni i pokazana je uspešna

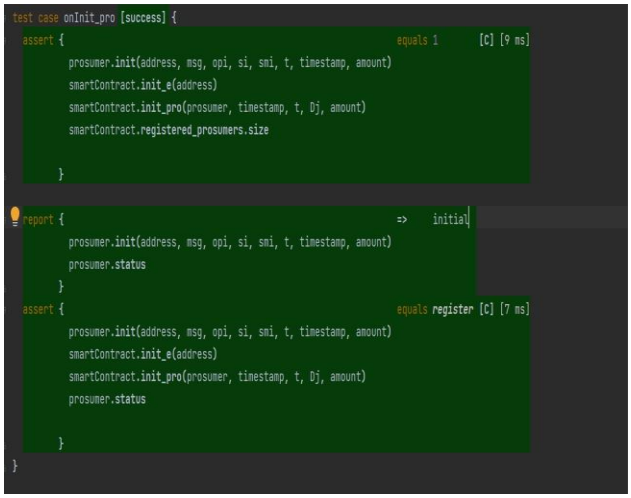
funkcionalnost trenutnog jezika. Korišćeno je više test slučajeva (slike od 5 do 8).



Slika br.6 *Test Consumer-a na događaj sendTx()*



Slika 7. *Test Consumera-a na događaj transfer()*



Slika 8. *Prikaz testa Smart Contract-a na događaj init_pro()*

6. ZAKLJUČAK

U radu je opisano testiranje DSL-a (*Domain-specific language*) – domenski specifičnog jezika za *energy* P2P (*Peer-to-Peer Service*) razvijenog u MPS okruženju uz podršku jezika KernelF-a. Opisani su učesnici *Prosumers* i *Consumers* kao i *Smart Contract* sa pripadajućim testovima i funkcionalnostima. Objašnjene su i implemetirane funkcionalnosti za učesnike, njihove događaje i stanja kao i za pametne ugovore.
Prednost razvoja domenskog jezika pomoću KernelF-a je u tome što su osnovni koncepti već implementirani u vidu *plugin*-a za jezik, a neophodno je samo konfigurisati okruženje i pravilno iskoristiti definisane koncepte. Takođe, omogućeno je veoma brzo izvršavanje i provera validnosti pomoću interpretera testova kojeg je neophodno konfigurisati. Prednost KernelF-a je u tome što je dobar za manje projekte kojima još osnovni koncepti nisu definisani, a moguće ih je veoma brzo testirati ili za jezgra velikih jezika izgrađenih na vrhu KernelF-a.

Mana KernelF-a je u tome što je nov jezik koji je i dalje u razvoju i zbog toga ne postoji puno primera i literature kao pomoći pri implementaciji jezika i generatora ili interpretera koda u MPS-u. Takođe, MPS je vezan za javu, pa se stoga celokupna aplikacija pokreće preko javine virtuelne mašine što smanjuje performanse.

Ovim radom su opisani i testirani osnovni koncepti jezika, ali implementacija još nije završena u potpunosti. U daljem razvoju bilo bi potrebno implementirati i DSO i primeniti algoritam za izračunavanje cena, nakon čega bi bila u potpunosti implementirana „menjačnica za energiju“. Posle ovoga bi se mogao izgenerisati kod za željenu platformu.

7. LITERATURA

[1] Smart Contracts dostupno na: https://www.ibm.com/topics/smart-contracts.
[2] Blockchain dostupno na: https://www.investopedia.com/terms/b/blockchain.asp
[3] Markus Vöelter, Design, evolution and use of KernelF
[4] Voluntis Healtcare dostupno na: https://www.voluntis.com/solutions/
[5] P2P Service dostupno na: https://www.investopedia.com/terms/p/peertopeer-p2p-service.asp
[6] Jae Geu Song, Eung soun Kang, Hyeon Woo Shin and Ju Wook Jang, A Smart Contract-Based P2P Energy trading Syistem with Dynamic Pricing on Etherum Blockchain
[7] Chekired, D.A.; Khoukhi, L.; Mouftah, H.T.; Decentralized cloud-SDN architecture in smart grid: A dynamic pricing model. IEEE Trans. Ind. Inform. 2018, 14, 1220–1231
[8] Markus Vöelter, KernelF- an Embeddable and Extensible Functional Language reference
[9] Marija Borisov, Jezik specifičan za domen za energy P2P trading baziran na MPS mbddr i KernelF specifikaciji, Ispitni rad iz Odabranih poglavlja savremenih metoda razvoja softvera (predmet na toktorskim studijama), 2021

Kratka biografija:

Marko Ercegovac rođen je u Sremskoj Mitrovici, Republika Srbija, 1997 god. Osnovne akademske stududije je upisao na Fakultetu tehničkih nauka Univerziteta u Novom Sadu 2016. Diplomirao je 2020. god.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.313
DOI: https://doi.org/10.24867/15BE10Stojanac

UPRAVLJANJE PRETVARAČIMA SA 2 I VIŠE NIVOA U ULOZI SINHRONVERTORA – INVERTORA KOJI OPONAŠAJU RAD SINHRONIH GENERATORA

CONTROL OF 2 LEVEL AND MULTILEVEL CONVERTERS IMPLEMENTED INTO SYNCHONVERTERS – INVERTERS THAT MIMIC SYNCHRONOUS GENERATORS

Nemanja Stojanac, Marko Vekić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Ovaj rad bavi se upravljanjem pretvaračima energetske elektronike (sa naglaskom na kaskadnim H – mostovima) u ulozi sinhronvertora.*

Ključne reči: *Kaskadni H – mostovi, sinhronvertor, Matlab Simulink*

Abstract – *This paper is about control of power electronic converters (with the focus on Cascaded H – bridges) implemented into syncrhonverters.*

Keywords: *Cascaded H – bridges, synchronverters, Matlab Simulink*

1. UVOD

Težnja za smanjenjem emisije ugljen-dioksida u atmosferu dovodi do usmeravanja razvoja elektro-energetskih sistema ka obnovljivim izvorima energije kao što su sunčeva svetlost (fotonaponski paneli) i vetar (vetrogeneratori). U tradicionalnom elektroenergetskom sistemu preovlađuju sinhroni generatori čije obrtne karakteristike igraju značajnu ulogu u balansiranju zahteva potrošača. Zbog tromosti rotora, prelazni procesi će biti blagi i bez ugrožavanja stabilnosti sistema.

Mana obnovljivih izvora je što u sebi sadrže pretvarače energetske elektronike koji su neinertni elementi. Zbog povećanog broja obnovljivih izvora energije inertnost celog sistema opada, što dovodi u pitanje stabilnost sistema. Neinertnost dovodi do naglih promena učestanosti sistema pri manjim promenama u aktivnoj snazi.

Jedan od načina da se prevaziđe nedostatak inertnosti u sistemu je da se pretvarači energetske elektronike prilagode radu koji će oponašati rad sinhronih mašina. U radu je, kao jedna od mogućnosti, opisana primena sinhronvertora [1].

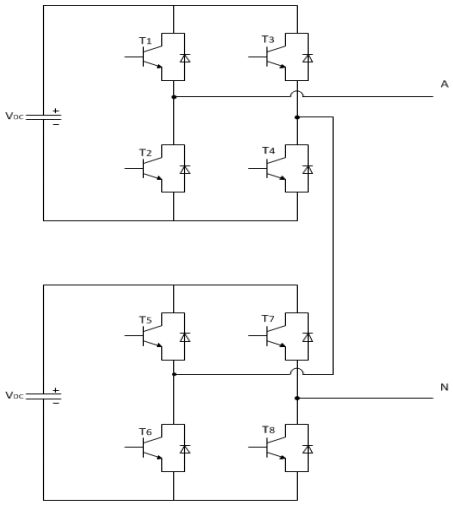
2. KASKADNI H – MOSTOVI (CHB)

Kaskadni H – mostovi spadaju u modularne pretvarače sa više nivoa. Modularne pretvarače odlikuje veći broj modula (ćelija) koji se međusobno povezuju [2]. Prednost modularnih pretvarača sa više nivoa u odnosu na ostale topologije je ostvarivanje značajnijih napona korišćenjem komponenata za manje napone, kao i jednostavna zamena neispravnih ćelija u slučaju kvara.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Marko Vekić, vanr. prof.

2.1. Topologija CHB

Jedan kaskadni H – most prikazan je na slici 1.

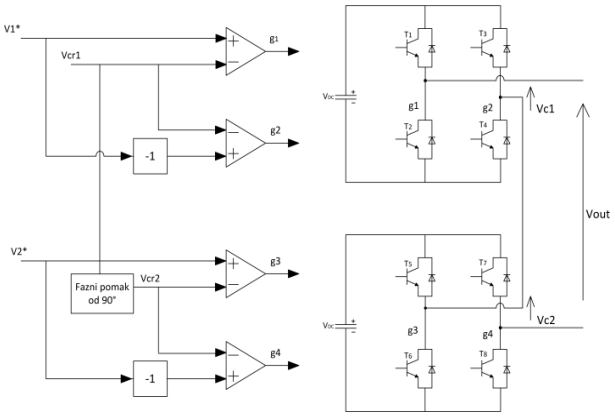


Slika 1. *CHB sa 5 nivoa*

Jedan H – most (jedna ćelija) može da ostvari 3 nivoa: $+V_{DC}$, 0, $-V_{DC}$. Ukoliko obe ćelije CHB-a sa slike 1 doprinose pozitivnom naponu, ostvariće se naponski nivo $+2V_{DC}$. Sličnim kombinovanjem izlaznih napona dve ćelije ovakav CHB ostvaruje ukupno 5 različitih naponskih nivoa: $+2V_{DC}$, $+V_{DC}$, 0, $-V_{DC}$, $-2V_{DC}$.

2.2. Fazno pomereni PWM (PS PWM)

Od mnoštva tehnika modulacije pretvarača sa više nivoa biće opisana fazno pomerena impulsno širinska modulacija (PS PWM, Phase Shifted Pulse Width Modulation).



Slika 2. *Implementacija PS PWM*

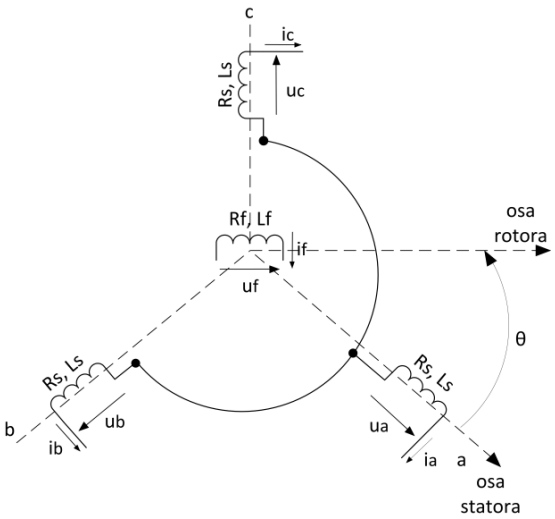
Svaka ćelija je modulisana nezavisno poređenjem modulišućeg i nosećeg signala. Modulišući signal je isti za sve ćelije iste faze, dok je noseći signal fazno pomeren za 180 deg/n, gde je n broj ćelija jedne faze. Hardverska implementacija data je na slici 2. Izlazni napon V_{out} jednak je zbiru napona na izlazu ćelija.

3. SINHRONVERTOR

Kombinovanjem energetske efikasnosti energetskih pretvarača u odnosu na sinhrone generatore, kao i osobine inertnosti sinhronih generatora, sinhronvertori postižu bilans aktivne snage (frekvencije) na efikasan način, bez ugrožavanja stabilnosti sistema. Pored toga, sinhronvertori mogu imati i ulogu STATCOM-a (statički sinhroni kompenzator) i učestvovati u kompenzaciji reaktivne snage (popravke sačinioca snage) ili obavljati nadzor napona zajedničke sabirnice. Načela rada sinhronvertora mogu se primeniti kako na pretvarače energetske elektronike sa dva nivoa, tako i na pretvarače s više nivoa.

3.1. Matematički model sinhronog generatora

Šema sinhronog generatora data je na slici 3.



Slika 3. *Šema sinhronog generatora [3]*

Matematički model se sastoji iz izraza za naponsku ravnotežu statora i izraza za fluksne obuhvate statora i rotora, kao i Njutnove jednačine kretanja:

$$\underline{u} = -R_s\underline{i} - \frac{d\underline{\Phi}}{dt} = -R_s\underline{i} - L_s\frac{d\underline{i}}{dt} + \underline{e} \qquad (1)$$

$$\begin{aligned} \underline{\Phi} &= L_s\underline{i} + M_f i_f \,\widetilde{\cos}\,\theta \\ \Phi_f &= L_f i_f + M_f(\underline{i}^T \times \widetilde{\cos}\,\theta) \end{aligned} \qquad (2)$$

$$\begin{aligned} \Phi_a &= L_s i_a - M i_b - M i_c + M_{af} i_f \\ \Phi_b &= -M i_a + L_s i_b - M i_c + M_{bf} i_f \\ \Phi_c &= -M i_a - M i_b + L_s i_c + M_{cf} i_f \\ \Phi_f &= M_{af} i_a + M_{bf} i_b + M_{cf} i_c + L_f i_f \end{aligned} \qquad (3)$$

gde su:

- $\underline{\Phi} = [\Phi_a\ \Phi_b\ \Phi_c]^T$
- $\underline{i} = [i_a\ i_b\ i_c]^T$
- $\underline{u} = [u_a\ u_b\ u_c]^T$
- $\underline{e} = [e_a\ e_b\ e_c]^T$
- $\widetilde{\cos}\,\theta = \left[\cos\theta\ \cos\left(\theta - \frac{2\pi}{3}\right)\ \cos(\theta - \frac{4\pi}{3}\right]^T$
- M- međusobna induktivnost statorskih namotaja
- L_s - samoinduktivnost statorskih namotaja
- M_{af}, M_{bf}, M_{cf} - međusobne induktivnosti faznih namotaja statora i namotaja rotora
- L_f - samoinduktivnost rotorskog namotaja
- i_a, i_b, i_c, i_f - fazne struje statora i rotora

Izraz za kontra-elektromotornu silu glasi:

$$\underline{e} = M_f i_f \omega\,\widetilde{\sin}\,\theta - M_f\frac{di_f}{dt}\widetilde{\cos}\,\theta \qquad (4)$$

gde je: $\widetilde{\sin}\,\theta = \left[\sin\theta\ \sin\left(\theta - \frac{2\pi}{3}\right)\ \sin(\theta - \frac{4\pi}{3}\right]^T$

Njutnova jednačina kretanja glasi:

$$J\frac{d\omega}{dt} = M_m - M_e - D_p\omega \qquad (5)$$

Veličina J predstavlja moment inercije, Mm moment opterećenja, Me elektromagnetni moment, dok D_p predstavlja faktor prigušenja (koeficijent trenja kod fizičkog generatora).

Izraz za elektromagnetni moment dobija se iz izraza za energiju sadržanu u magnetnom kolu:

$$M_e = -\frac{\partial W}{\partial \theta_m} \qquad (6)$$

Tako je izraz za elektromagnetni moment jednak:

$$M_e = pM_f I_f\left(\underline{i}^T \times \widetilde{\sin}\,\theta\right) \qquad (7)$$

Za upravljanje sinhronvertorom bitni su još i izrazi za aktivnu i reaktivnu snagu koji se dobijaju kao:

$$\begin{aligned} P &= \underline{i}^T \times \underline{e} \\ Q &= -\underline{i}^T \times \underline{e_g} \end{aligned} \qquad (8)$$

gde $\underline{e_g} = M_f i_f \omega\,\widetilde{\cos}\,\theta$ predstavlja signal amplitude kao elektromotorna sila, ali je zakašnjen za 90^0 .

Krajnji izrazi za aktivnu i reaktivnu snagu slede:

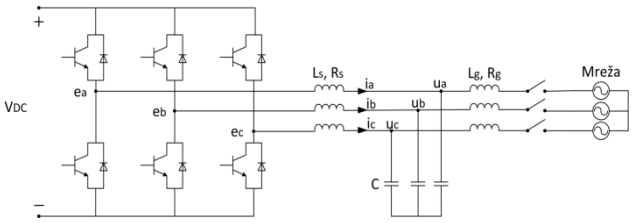
$$\begin{aligned} P &= \omega M_f i_f(\underline{i}^T \times \widetilde{\sin}\,\theta) \\ Q &= -\omega M_f i_f(\underline{i}^T \times \widetilde{\cos}\,\theta) \end{aligned} \qquad (9)$$

3.2. Primena invertora u ulozi sinhronvertora

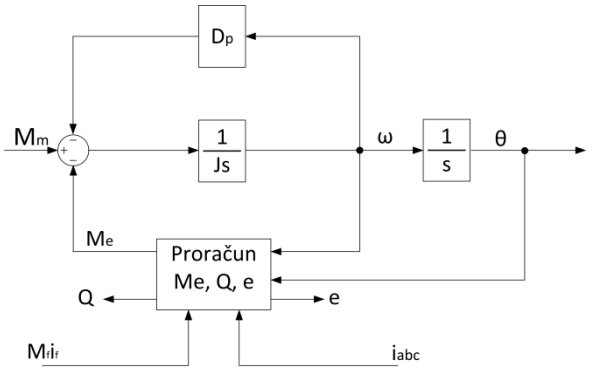
Sinhronvertor se sastoji iz dva dela: energetike (slika 4) i upravljačke elektronike (slika 5).

Energetika se sastoji iz trofaznog invertora, LC filtra (sa uvaženim otpornostima), sprežnog RL filtra i prekidača za priključenje na mrežu.

Uloga LC filtra je kako da oponaša otpornosti i induktivnosti namotaja sinhrone mašine, tako i da filtrira prekidački napon kako bi se dobio približno sinusan oblik kao što je slučaj kod sinhrone mašine.



Slika 4. *Energetski deo sinhronvertoraž*



Slika 5. *Upravljačka elektronika*

Upravljačka elektronika se sastoji iz Njutnove jednačine kretanja i sklopova za proračun elektromagnetnog momenta, reaktivne snage i reference elektromotorne sile. Ulazi u upravljački slop su moment opterećenja i virtuelni fluks rotora M_fI_f, dok promenljive stanja su struje statora, virtuelna ugaona brzina i virtuelni ugao.

3.3. Dejstvo sinhronvertora

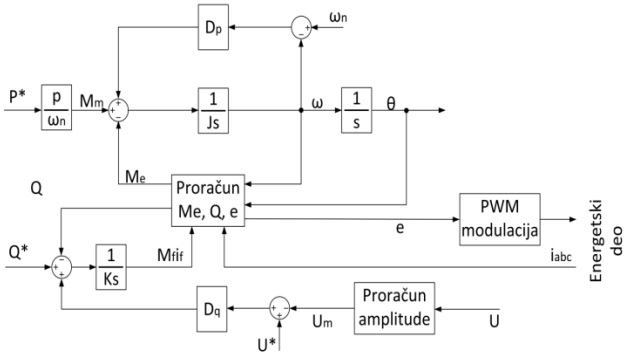
Kao što je već rečeno, sinhronvertorom se želi postići inertan rad pretvarača energetske elektronike. U nastavku su opisana dva bitna dela za upravljanje sinhronvertora, a to su frekventna (Pf) i naponska (QV) droop kontrola.

3.1.1. Pf regulacija

Pf regulacija se ostvaruje jednostavno proširenjem upravljačke elektronike sinhronvertora. Za droop kontrolu može se iskoristiti faktor prigušenja koji se proračunava kao odnos promene mehaničkog momenta usled promene virtuelne ugaone učestanosti:

$$D_p = \frac{\Delta M_m}{\Delta \omega} = \frac{\Delta M_m}{M_{mn}} \frac{\omega_n}{\Delta \omega} \frac{M_{mn}}{\omega_n} \tag{10}$$

Izmenjeni upravljački sklop prikazan je na gornjem delu slike 6. Mehanički moment zadaje se indirektno zadavanjem aktivne snage.



Slika 6. *Blok algoritam rada sinhronvertora*

Moment inercije bira se spram proračunatog faktora prigušenja i željene vremenske konstante frekventne petlje:

$$J = D_p \tau_f \tag{11}$$

3.1.2. QV regulacija

Algoritam QV regulacije prikazan je na donjem delu slike 7. Zadaju se reaktivna snaga i amplituda faznog napona.

Faktor D_q predstavlja analogiju faktoru prigušenja D_p, dok koeficijent K predstavlja analogiju momentu inercije.

Faktor D_q se proračunava spram odnosa promene reaktivne snage po promeni napona:

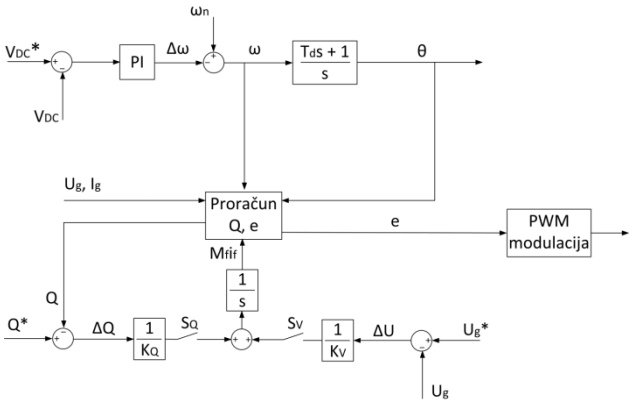
$$D_q = \frac{\Delta Q}{\Delta V} = \frac{\Delta Q}{Q_n} \frac{V_n}{\Delta V} \frac{Q_n}{V_n} \tag{12}$$

Koeficijent K se proračunava spram faktora D_q i vremenske konstante naponske petlje.

$$K = \omega_n D_q \tau_v \tag{13}$$

4. STATCOM-ski rad sinhronvertora

Sinhronvertor se može prilagoditi STATCOM-skom režimu rada. Potrebno je izmeniti upravljačku elektroniku dok energetski deo zadržava isti izgled (slika 7).



Slika 7. *Upravljačka elektronika sinhrovertora u STATCOM-skom režimu*

STATCOM-ski rad podrazumeva kontrolu napunjenosti kondenzatora jednosmernog kola putem razmene aktivne snage (podešavanjem virtuelnog ugla θ), kao i regulaciju reaktivne snage odnosno napona sabirnice zajedničkog priključenja putem zadavanja virtuelnog fluksa (elektromotorne sile).

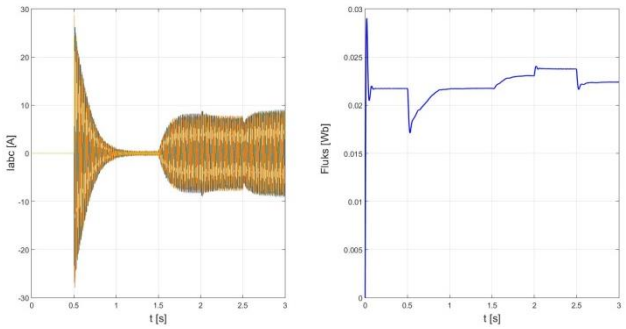
5. SIMULACIJE SINHRONVERTORA

Simulacije su vršene u Matlab Simulink okruženju.

5.1. Pf regulacija

Faktor prigušenja proračunat je tako da se za 0,5% promene učestanosti aktivna snaga (moment) menja za 100%.

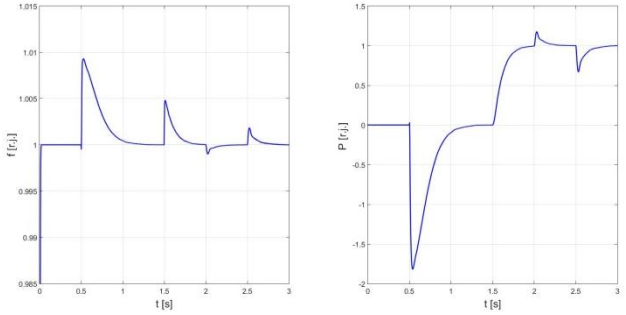
Izgled faznih struja i virtuelnog fluksa dat je na slici 8.



Slika 8. *Fazne struje (levo) i virtuelni fluks (desno)*

Izgled faznih struja identičan je strujama statora sinhronog generatora.

Na slici 9 prikazani su učestanost i aktivna snaga.

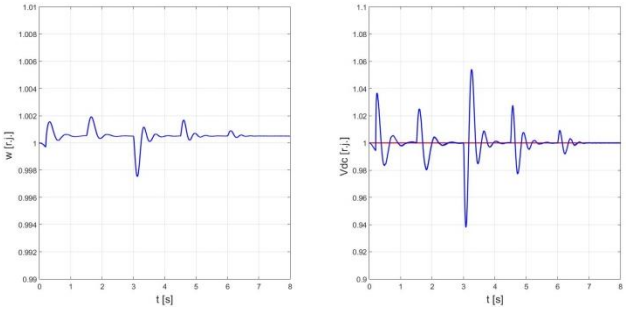


Slika 9. *Učestanost (levo) i aktivna snaga (desno)*

U 0,5s sinhronvertor se priključuje na mrežu. Dolazi do propada snage i porasta frekvencije, koji se u kratkom roku vraćaju na zadate vrednosti. U trenutku 1,5s zadaje se mehanička snaga od 1 r.j, kad dolazi do porasta frekvencije za 0,5%. Posle izvesnog vremena snaga dostiže zadatu vrednost dok se frekvencija vraća na nominalnu.

5.2 STATCOM-ski režim

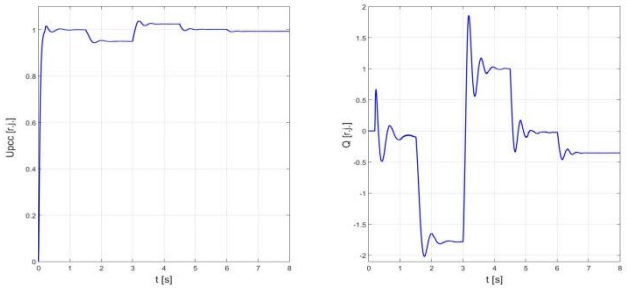
Na slici 10 prikazani su odzivi virtuelne ugaone brzine i napona na kondenzatoru jednosmernog kola. Uočava se da se DC napon kreće okvirima zadate vrednosti.



Slika 10. *Ugaona brzina (levo) i DC napon (desno)*

Na slici 11 prikazani su odzivi napona sabirnice zajedničkog priključenja i reaktivne snage. Sve do 3s aktivan je prekidač S_V čime je STATCOM u režimu regulacije napona, prvo na nominalnu vrednost a zatim na vrednost 95%. Nakon 3s STATCOM prelaži u režim regulacije reaktivne snage na zadatu vrednost od 100%.

Nakon 4,5s nastupa droop režim u kom su aktivna oba prekidača istovremeno, te se napon i reaktivna snaga menjaju po droop karakteristici spram zadatih vrednosti.



Slika 11. *Napon PCC (levo) i reaktivna snaga (desno)*

5. ZAKLJUČAK

Kako je potreba za veštačkom inercijom nastala nedavno sa rastom obnovljivih izvora sinhronvertori su još uvek fazi istraživanja. Ono što je sigurno je da će uređaji poput sinhronvertora biti predmet istraživanja u narednim godinama i da će naći svoju ulogu u rešavanju izazova nadolazećih mikromreža i pametnih mreža.

6. LITERATURA

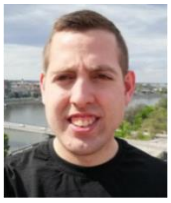
[1] Q.-C Zhong and G. Weiss, "Synchronverters: Inverters that mimic synchronous generators" IEEE Trans. Ind. Electron., vol. 58, no. 4, pp. 1259-1267, Apr. 2011.

[2] M. Marchesoni, M. Mazzucchelli, and S. Tenconi, "A nonconventional power converter for plasma stabilization," *IEEE Transactions on Power Electron*, vol. 5, no. 2, pp. 212–219, Apr 1990.

[3] A. E. Fitzgerald, C. Kingsley, "Electric Machinery", vol 6, 1962.

[4] J. Rodriguez, L. Franquelo, S. Kouro, J. Leon, R. Portillo,M. Prats, and M. Perez, "Multilevel converters: An enabling technology for high-power applications," *Proceedings of the IEEE*, vol. 97, no. 11, pp. 1786–1817, Nov 2009.

Kratka biografija:



Nemanja Stojanac rođen je u Novom Sadu 1995. god. 2014. godine upisuje osnovne akademske studije na Fakultetu tehničkih nauka u Novom Sadu, smer – energetika, elektronika i telekomunikacije. Nakon završenih osnovnih studija 2020. godine upisuje master akademske studije na Fakultetu tehničkih nauka u Novom Sadu.

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

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/15BE11Mihajlovic

PROJEKTOVANJE UNIVERZALNE VERIFIKACIONE KOMPONENTE ZA APB PROTOKOL PRIMJENOM UVM METODOLOGIJE

DEVELOPING UNIVERSAL VERIFICATION COMPONENT FOR APB PROTOCOL USING UVM METODOLOGY

Jovana Mihajlović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Zadatak rada jeste projektovanje verifikacione komponente za APB protokol korišćenjem UVM metologije. Programski jezik je SystemVerilog. U radu je dat uvod u SystemVerilog jezik i UVM metodologiju. Na kraju su prikazani rezultati testiranja i funkcionalni coverage.*
Ključne reči: *UVM, SystemVerilog, APB protocol, UVC, functional verification*
Abstract: *The task of this paper is developing the verification component for APB protocol using UVM metodology. Programming language is SystemVerilog. The paper contains inroduction into SystemVerilog and UVM. Also, results of test scenarios and functional coverage are presented.*
Ključne reči: *UVM, SystemVerilog, APB protocol, UVC, functional verification*

1. UVOD

Pri projektovanju savremenih digitalnih sistema, kao posljedica povećanja njihove kompleksnosti funkcionalna verifikacija je postala jedan od najzahtjevnijih koraka.
Funkcionalna verifikacija ima zadak da provjeri funkcionalnu tačnost dizajna u odnosu na specifikaciju. Pri tome, funkcionalna verifikacija se ne bavi tačnošću specifikacije koja se već smatra tačnom.
Tokom poslednjih godina uvedeni su novi standardi i metodologije za verifikaciju, kao što je UVM metodologija (Universal Verification Methodology).
U ovom master radu će biti prikazana primjena UVM metodologije na primjeru projektovanja univerzalne verifikacione komponente za APB (Advanced Peripheral Bus) protokol. Cilj rada je izrada verifikacione komponente koja se kasnije može upotrijebiti prilikom verifikacije proizvoljnog sistema sa AMBA (Advanced Microcontroller Bus Architecture) magistralom koji koristi APB protokol. Univerzalna verifikaciona komponenta će biti opisana korišćenjem SystemVerilog jezika.
Specifičnosti SystemVerilog jezika koji se koristi i u verifikaciji i za dizajn biće predstavljene u drugom poglavlju.
U ovom radu biti će predstavljena i sama UVM metodologija.
U četvrtom poglavlju će biti opisan APB protokol.
Nakon toga će biti predstavljena APB UVC komponenta i rezultati testiranja.

NAPOMENA:
Ovaj rad proistekao je iz master rada, čiji mentor je bio prof. dr Darko Marčetić.

2. SystemVerilog

SystemVerilog predstavlja prvi industrijski HDVL (Hardware Design and Verification Language) jezik. Koristi se kako za modelovanje i dizajn, tako i za verifikaciju u industriji poluprovodnika. On čini kombinaciju jezika za dizajniranje kao što su Verilog i VHDL sa mogućnostima jezika specijalizovanih za verifikaciju (OpenVera, e), uz uobičajene programske jezike C i C++. Zasnovan je na Verilogu.
2005. godine postaje IEEE (Institute of Electrical and Electronics Engineers) standard – 1800 IEEE, a 2009. je dodatno unaprijeđen.
Kako vrijeme prolazi, ovaj objektno orjentisani jezik pronalazi sve veću primjenu u RTL (Register Transfer Level) dizajniranju, verifikaciji zasnovanoj na assertionima kao i u izgradnji okruženja za constrained random verifikaciju.
Operatori u SystemVerilog-u su mješavina Verilog i C operatora. U oba jezika tip i veličina operatora su fiksni, pa je tako i u SystemVerilog-u.
Za provjeru protokola i ponašanja sistema u SystemVerilog jeziku se koriste assertion-i. Oni se uglavnom koriste za provjeru dizajna, ali mogu biti korišćeni i za prikupljanje functional coverage-a.
Njihova prednost je što umanjuju vrijeme potrebno za debug i brže dovode do njihovog otkrivanja, imaju mogućnost interagovanja sa C funkcijama i imaju podesiv "severity" ili nivo značaja.
Mogu biti korišćeni za:

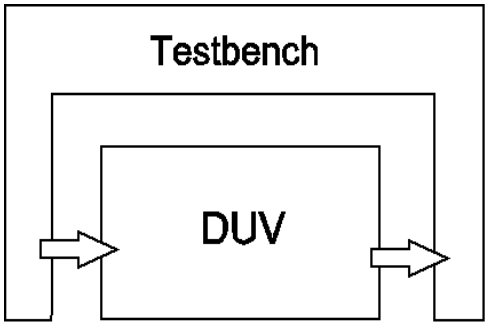
- provjeru nekih logičkih uslova
- provjeru ispravnosti sekvenci korišćenjem vremenskih izraza "temporal expressions"
- provjeru nedefinisanih stanja signala (X,Z)
- "onehot" provjera …

Assertions mogu da se podijele u dvije grupe: "immediate assertions" i "concurrent assertions".

3. UVM METODOLOGIJA

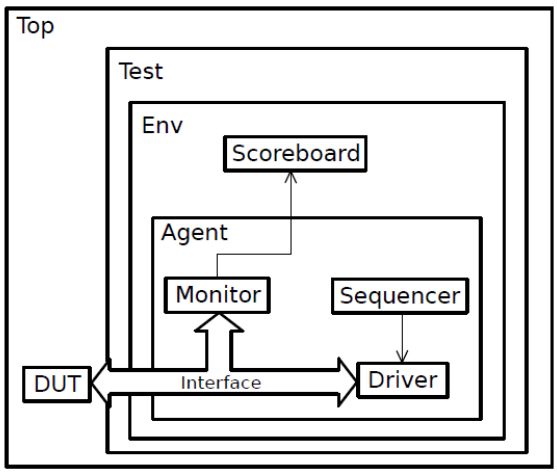
Prilikom dizajniranja elektronskih kola se koriste HDL (Hardware Description Language) jezici, od kojih su najpoznatiji Verilog i VHDL, koji omogućavaju da se apstraktni modeli i tražena funkcionalnost prevedu na „pravi" hardware. Međutim kada se ovaj deo posla završi, postavlja se pitanje da li implemetirano zaista radi ono što je specifikacijom predviđeno. Odgovor treba da da proces verifikacije, čiji početak u praksi nikada ne čeka kraj dizajniranja, već se ta dva procesa odvijaju paralelno.
Proces verifikacije se obavlja kreiranjem virtuelnog okruženja oko dizajna (DUV – Device Under Verifica-

tion), koje omogućava slanje stimulusa i analizu njegovog ponašanja na isti. Ovo okruženje se naziva *testbench.*



Slika.3.1 DUV i Testbench

Proces „funkcionalne verifikacije“ zahteva da se u testbench implementira ista funkcionalnost koja se očekuje od dizajna i da se na osnovu poređenja ponašanja utvrdi njegova ispravnost.
Tradicionalni pristup je bio pisanje direktnih testova, koji proveravaju neku tačno određenu situaciju. Ovakav način testiranja ima linearno povećanje potrebnog vremena sa povećanjem kompleksnosti dizajna. Kako kompleksnost elektronskih kola raste iz godine u godinu vreme za verifikaciju ovom metodom više nije prihvatljivo i bilo je neophodno osmisliti način koji će moći brže da dovede do rezultata.
Rešenje je pronađeno u obliku „constraint random verification“ metoda, koje se zasnivaju na slanju nasumičnih stimulusa u okviru nekih, unaprijed isplaniranih ograničenja. Da bi ovo bilo omogućeno, neophodno je automatizovati proces generisanja stimulusa, kao i blokova koji će pratiti pruozrokovane izlaze, blokova koji predviđaju kakvi izlazi treba da budu i blokova koji će vršiti analizu i poređenje.
Takođe, zbog same prirode nasumičnog saobraćaja, neophodno je i nekako pratiti koje situacije su testirane.
Još jedna prednost constraint random verification metoda jeste to što generisanjem nasumičnog saobraćaja omogućavaju pogađanje situacija koje eventualno nisu predviđene, a koje se mogu ispostaviti kao značajne.
Da bi bilo omogućeno ponovno korišćenje testbencha sa minimalnim promenama pri prelasku sa verifikacije jednog, na drugi dizajn, neophodno je standardizovati implementaciju svih neophodnih blokova.
Zasnovan na objektno-orjentisanom jeziku, SystemVerilog-u, UVM poseduje čitav niz klasa koje omogućavaju implementaciju cijelog testbench-a, koji će se od projekta do projekta uglavnom razlikovati samo po broju instanciranih UVC-eva (Universal Verification Component) i po dodavanju koda koji je specifičan za konkretnu specifikaciju/protokol koji je potrebno verifikovati.
Svi blokovi koji se koriste pri izgradnji testbench-a nastaju korišćenjem klasa koje su izvedene od neke od osnovnih, i dodavanjem koda koji će je učiniti primenljivom za konkretan slučaj zadate specifikacije.
Klase mogu biti podeljene u tri glavne grupe, uvm_component koje su korišćene za izgradnju hijerarhiske strukture testbench-a, uvm_object koje se koriste za strukture podataka za konfigurisanje testbench-a, i uvm_transaction koje se koriste za generisanje stimulusa i analizu i prikupljanje coverage-a.



Slika.3.2 Struktura testbench-a

Svaka od ovih klasa već ima ugrađene metode u skladu sa namjenom za koju su predviđene, a moguće je dodavanje novih u izvedenim klasama.
„Top level“ klasa u UVM testbench-u je označena kao test klasa, i ona je odgovorna za konfigurisanje testbench-a, iniciranje izgradnje sledećeg nivoa u hijerarhiji i iniciranje slanja stimulusa pokretanjem glavne sekvence. Uobičajeno nastaje nasleđivanjem test_base klase.
Sama struktura UVM testbench-a je modularno izgrađena da bi omogućila ponovno iskorišćenje koda i u drugim projektima, „horizontal reuse“, ili i u višem nivou hijerarhije unutar istog projekta, „vertical reuse“. Dvije osnovne komponente koje omogućavaju ponovno korišćenje su samo okruženje – env, i agent (koji bi trebalo da opisuje ponašanje nekog protokola ili modula).

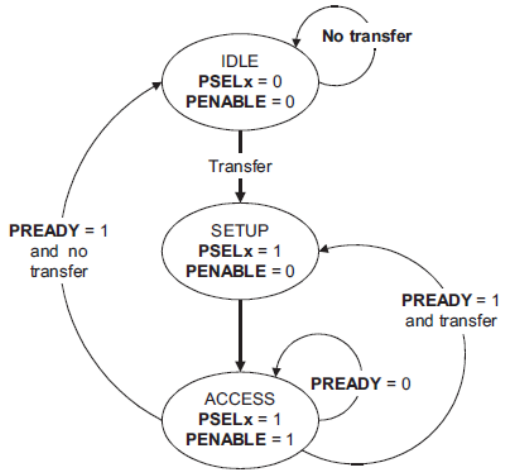
4. APB PROTOKOL

Advanced Peripheral Bus (APB) je dio Advanced Microcontroller Bus Architecture (AMBA) familije protokola. On definiše low-cost interfejs koji je optimizovan za minimalnu potrošnju energije i predstavlja interfejs smanjene kompleksnosti.
APB ne posjeduje pipeline, i predviđen je za low-bandwidth periferije koje ne zahtjevaju visoke performanse AXI protokola.
Da bi se olakšala integracija APB periferija u bilo koji dizajn, on je predviđen da na svaku uzlaznu ivicu takta obavlja po jednu promjenu. Za svaki transfer su neophodna bar dva takta.
APB može da se poveže sa:

- AMBA Advanced High-performance Bus (AHB)
- AMBA Advanced High-performance Bus Lite (AHB-Lite)
- AMBA Advanced Extensible Interface (AXI)
- AMBA Advanced Extensible Interface Lite (AXI4-Lite)

Kao primjer, može se koristiti za pristup programabilnim kontrolnim registrima perifernih uređaja.
Sam protokol je unaprijeđen par puta u odnosu na prvu verziju predstavljenu 1998. godine, a verzija koja će biti korišćena u ovom radu je verzija 2.0, koja u trenutku pisanja predstavlja najnoviju verziju.
APB protokol poseduje dve nezavisne magistrale podataka, od kojijh se jedna koristi za čitanje, a druga za upis podataka. Magistrale mogu biti maksimalne širine 32

bita. Pošto poseduju zajedničke handshake signale, nije moguće istovremeno prosleđivanje podataka na obje strane.



Slika 4.1 Prelazi mašine stanja

Mašina stanja prolazi kroz naredna stanja:

- IDLE - Ovo je inicijalno stanje, u koje se prelazi kada nema transfera.
- SETUP - Iniciranje transfera inicira prelaz u ovo stanje. U ovom stanju se postavlja odgovarajući selekcioni signal, PSELx. Magistrala ostaje u ovom stanju samo jedan ciklus takta i odmah zatim prelazi u ACCESS stanje na sledeću uzlaznu ivicu takta.
- ACCESS - Signal dozvole, PENABLE, se postavlja tokom ovog stanja. Adresa, kontrolni signali i podatak za upis moraju biti stabilni tokom ovog prelaza iz SETUP stanja.
- Prelaz iz ovog stanja se kontroliše signalom PREADY:
- Ukoliko je PREADY na niskom nivou, magistrala ostaje u ACCESS stanju.
- Ukoliko slave uređaj postavi PREADY na visoku vrijednost, onda se izlazi iz ACCESS stanja i prelazi u IDLE stanje ukoliko nema više zahtjeva za transferima. Ako postoji zahtjev za novim transferom, prelazi se direktno u SETUP stanje.

5. PROJEKTOVANJE UNIVERZALNE VERIFIKACIONE KOMPONENTE

Agent je osnovna komponenta u kojoj se kreiraju sve podkomponente:

- konfiguracija,
- interface,
- monitor,
- driver,
- sequencer.

Za prosleđivanje konfiguracije podkomponentama je iskorišćen konfiguracioni objekat.

hdh_apb_config m_cfg;

On je postavljen u bazu podataka od strane neke komponente više u hijerarhiji (u ovom konkretnom primjeru, u env komponenti), a ovdje se dohvata zarad konfigurisanja samog agenta.

Ukoliko konfiguracija svakog agenta (master i slave) nije setovana, podrazumijevaće se default konfiguracija.

Interface nije predstavljen na slici 5.1 budući da predstavlja komponentu koju je potrebno instancirati odvojeno u testbench-u. Nakon toga se dodijeli određenom agentu, te se na taj način povezuju.

virtual hdh_apb_if HDH_APB;

Pošto je predviđeno da ovaj UVC osim monitorisanja ispravnosti saobraćaja (pasivan UVC) bude u mogućnosti i da šalje saobraćaj (aktivan UVC), osim monitor-a neophodno je kreirati i driver i sequencer komponente.

Za sve agente u verifikacionom okruženju (VE), odgovarajući konfiguracioni objekat (tipa hdh_apb_config class) se kreira u build fazi. Isti objekat se koristi i za master i za slave agenta. Nakon što su konfiguracioni objekti kreirani, konfiguracionim poljima se dodijeljuju određene vrijednosti.

U virtuelni interface su dodati signali definisani protokolom.

Pomoću define-ova su definisane širine signala na magistrali. Oni se postavljaju u hdh_apb_types.svh fajlu.

Interface sadrži dva input porta: PRESETn i PCLK. Potrebno je da testbench drive- uje PRESETn kao active low kako bi saobraćaj bio moguć.

Za monitorisanje APB magistrale kreirane su dvije monitor komponente – master i slave monitor. Master monitor provjerava ispravnost signala koje slave postavlja i obrnuto, slave monitor provjerava ispravnost signala koje master postavlja i kontroliše promijene vrijednosti na magistrali.

Master monitor sačeka da PSEL bude postavljen, a zatim i PREADY . Pročita vrijednosti ostalih signala i upakuje ih u odgovarajuću transakciju. Transakcija se preko analysis port- a šalje u scoreboard.

Sve ovo se dešava u forever begin ... end petlji što omogućava hvatanje svih transakcija na magistrali.

Korišćenjem fork join petlje u paraleli se izvršavaju taskovi za provjeru ispravnosti saobraćaja.

Slave monitor sačeka da PSEL bude postavljen, a zatim provjerava da li vrijednost PSEL-a odgovara slave_id – u. Ukoliko odgovara, nastavlja se monitoring jer to znači da je on taj slave kojeg je master selektovao. Nakon toga sačeka da PREADY bude postavljen, te kao i master monitor pročita vrijednosti ostalih signala i upakuje ih u odgovarajuću transakciju. Transakcija se preko analysis port- a šalje u scoreboard.

U oba monitora se sve ovo dešava u forever begin ... end petlji što omogućava hvatanje svih transakcija na magistrali.

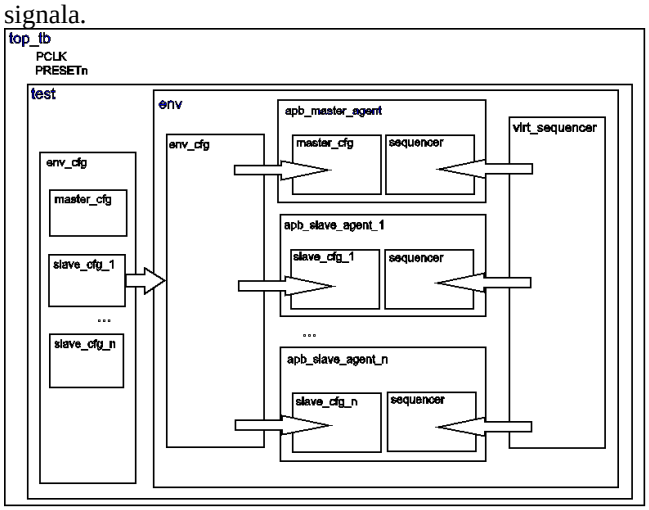
Korišćenjem fork join petlje u paraleli se izvršavaju taskovi za provjeru ispravnosti saobraćaja.

Kreiraju se dvije driver klase, jedna za master, druga za slave uređaje, gdje se pri samom kreiranju UVC-a u zavisnosti od konfiguracije kreira driver jednog ili drugog tipa.

U slučaju da uređaj radi kao master, prvo se inicijalizuje magistrala postavljanjem nekativnih vrijednosti kontrolnih signala. A zatim se u forever begin..end petlji dohvati sequence_item i pokrene samo postavljanje signala na magistralu u skladu sa protokolom.

Da bi uporedo bio podržan i reset sistema, prethodno opisan proces je ubačen u fork..join_any strukturu kao proces koji se izvršava u paraleli zajedno sa taskom

reset_signals. Ovaj task obavlja reset vrijednosti na žicama magistrale u slučaju da PRESETn signal bude postavljen na aktivnu vrijednost.
U slučaju kada driver radi kao slave, prvo se inicijalizuje vrijednost signala koje postavlja slave uređaj. A zatim se na sličan način kao na master strani u forever begin..end petlji u paraleli, korišćenjem fork..join_any strukture pokreću procesi za reset i za postavljanje signala na magistrali.
Pošto slave uređaj ne inicira saobraćaj, već praktično samo odgovara na zahtev od master uređaja, driver čeka da preko PSEL signala prepozna da je on slave kome je zahtjev poslat, a zatim u zavisnosti od tipa transakcije (read/write) odgovara na zahtjev. U slučaju write transakcije vrijednost sa PWDATA signala upisuje u asocijativni niz koji služi da simulira memoriju, i nasumično postavlja kontrolne signale kao odgovor.
U slučaju read transakcije, provjerava da li za zadatu adresu već ima unijet podatak u asocijatvni niz (što bi značilo da je u nekom prethodnom trenutku poslata write transakcija sa istom adresom) i ukoliko ima, tu vrijednost postavlja na PRDATA signale. Ukoliko ne postoji prethodni unos, na PRDATA signale postavlja nasumično izabran podatak (koji bi mogao da simulira vrijednost na memorijskoj lokaciji na koju nikada nije izvršen upis nekog smislenog podatka).
Da bi testiranje bilo omogućeno neophodno je postojanje testbench fajla. U njemu se napravi modul u kome bi bio instanciran DUT i povezani njegovi signali na virtuelni interface, ili u ovom slučaju u nedostatku pravog RTL modula kreira se samo virtuelni interface preko koga će biti povezani master i slave APB agenti.
Kreirani virtuelni interface se postavlja u bazu podataka da bi komponente niže u hijerarhiji mogle da ga dohvate.
U testbench- u se obavlja i generisanje clock i reset signala.



Slika 5.1. Dijagram okruženja za testiranje

Kako bi se pokrila funkcionalnost i omogućilo pogađanje situacija zadatih preko coverage- a, kreirani su testovi:

- u2u_read_test
- u2u_write_test
- u2u_slverr_test
- u2u_random_test
- u2u_multislave_test.

Svi testovi koriste hdh_apb_base_sequence.
Za pokretanje sekvenci u testu je neophodno kreirati samu sekvencu, obaviti dodjelu nasumičnih vrijednosti variablama i na kraju pozivom start metode pokrenuti sekvencu.
Pošto se coverage sakuplja samo kada postoji validna transakcija nije bilo moguće pogoditi bin PSEL = 0, iako je takva situacija očigledno postojala. Kada neka situacija nije pogođena u coverage-u, neophodno je to „opravdati“ u dokumentaciji inače se proces verifikacije ne može smatrati potpunim.
U ovom slučaju to je opravdano, i ova vrijednost bin-a bi mogla da bude isključena iz računanja konačnog coverage-a. Isključenje bin-a je ovdje uradjeno iz samog koda. Druga opcije jeste iskljucenje bin-a pomocu samog alata.
Takođe, iz koda je isključen bin cross coverage-a koji provjerava da li je tokom READ transfera PSTRB imao vrijedsnost koja nije nula. Opravdano je isključenje ovog bin-a budući da nije po protokolu. Druga opcija je da se napiše kao illlegal bin, te bi u slučaju narušavanja protokola simulacija bila završena. Pošto već imamo provjeru za PSTRB tokom READ transfera, ovdje to nije urađeno.

6. ZAKLJUČAK

U ovom radu dat je primjer izrade jedne reusable komponente primjenom UVM metodologije.
UVM metodologija predstavlja jedan od rezultata razvoja constrained random verifikacije. Random verifikacija omogućava lakše pokrivanje šireg spektra scenarija.
Ovakav pristup omogućava ubrzavanje procesa verifikacije, te je dominantan u poslednjih 20 godina.
APB UVC implementira APB protokol i sve njegove feature. Podržana je potpuna APB protokol specifikacija: jedan master i vise slave- ova, transakcije upisa i čitanja sa i bez čekanja, slave error response, zaštita pristupa, strobe signal. Definisanje metrike za praćenje napretka verifikacije omogućeno je coverage- em.
Funkcionalna pokrivenost definisana je u master komponenti. Rezultati coverage-a pokazuju 100% pokrivenost. Takođe, rezultati regresije pokazuju prolaznost svih testiranih scenarija.
Verifikacioni plan je urađen pomoću vPlan programa. U njemu se nalazi spisak svih test scenarija, coverage item-a i checkera. Svi item- i su mapirani u odnosu na specifikaciju.

7. LITERATURA

[1] www.verificationacademy.com
[2] https://www.elektronika.ftn.uns.ac.rs/funkcionalna-verifikacija-hardvera/
[3] www.doulos.com
[4] Universal Verification Methodology (UVM) 1.1 User’s Guide
[5] Universal Verification Methodology (UVM) 1.1 Class Reference
[6] SystemVerilog 3.1a Language Reference Manual
[7] AMBA APB Protocol Version 2.0 Specification

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.38:620.9
DOI: https://doi.org/10.24867/15BE12Acanski

ELEKTRIČNI TROTINETI MODELOVANJE I SIMULACIJA RADA

ELECTRIC SCOOTERS MODELING AND SIMULATION OF WORK

Filip Ačanski, Vladimir A. Katić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu razmatraju se električni trotineti. Počevši od istorije razvoja prvih električnih vozila, preko razmatranja dostupnih električnih vozila male snage. Takođe su razmatrane i zakonske regulative koje se tiču električnih vozila male snage kao i njihovi pogonski podsklopovi. Na kraju pristupilo se modelovanju odgovarajućeg električnog trotineta u programskom jeziku Matlab/Simulink tačnije modela NineBot es5. Cilj simulacije jeste da se što verodostojnije predstavi fizički model električnog trotineta.*

Ključne reči: *Električni trotineti, Simulacija, Modelovanje*

Abstract – *This paper deals with electric scooters. Starting with early history of development of electric vehicles, available models of electric vehicles of small power (LEV) on the market are considered. Legislations for electric scooters and small power vehicles in general are also considered. The main part of the paper is a simulation and modeling of an electric scooter using NineBot ES4 in Matlab/Simulink The goal is to show that model made in Matlab gives adecvate results of simulation similar or same to results in reality.*

Keywords: *Electric scooters, Modeling, Simulation*

1. UVOD

Proizvodnja električne energije i saobraćaj danas predstavljaju najdominatnije izvore CO_2 drugih štetnih materija. Obnovljivi izvori energije zajedno sa razvojem i primenom električnih vozila deo su novog plana za rešavanje ovog problema.

Istorija razvoja električnih vozila seže u daleku prošlost. Istorijski dokumenti, koji govore o modelu prvog električnog vozila datiraju iz 1881. godine. Tvorac tog vozila bio je Victor Trouve, francuski pronalazač.

Godine 1884. u Wolverhampton-u engleski pronalazač Thomas Parker kreira prvo električno vozilo, dok jedini materijalni dokaz predstavlja slika vozila načinjena 1885. godine [1].

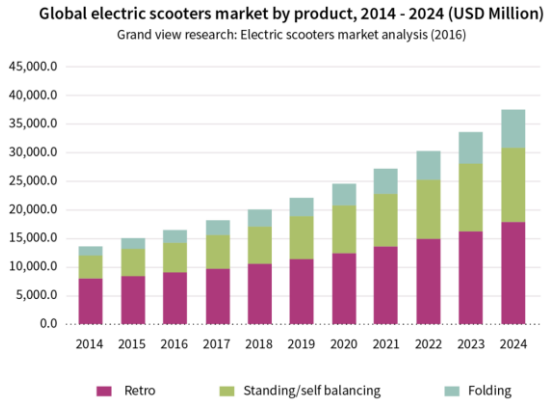
Međutim, primat na tržištu automobila početkom XX veka preuzimaju vozila sa SUS motorima. Neki od razloga, koji su doprineli ovom stanju su sledeći: slab razvoj infrastrukture za punjenje baterija, otkrića nalazišta nafte, veći domet koji su pružala vozila sa SUS motorima i dr.

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

Električna vozila zahvaljujući napretku na polju razvoja električnih baterija i tehnologije izrade komponenti energetske elektronike, ali i sve većoj zabrinutosti za životnu sredinu i uticaj čoveka na klimatske promene ponovo dobijaju na popularnosti 90-ih godina XX veka.

Kao paralelna kategorija, dolazi do naglog razvoja električnih vozila male snage ili LEV (*Light Electric Vehicles*), tačnije nominalne snage manje od 3 kW. Moderni oblik i dizajn, ova vozila poprimaju početkom XXI veka. U njih spadaju električni skuteri, električni trotineti, električni bicikli i dr. Perspektiva tržišta ovih vozila u svetu je veoma velika. Na slici 1 prikazana je godišnja prodaja i perspektive tržišta *LEV* vozila u svetu u periodu od 2014-2024 [2]. Može se uočiti velika vrednost i značajan rast, naročito novih rešenja: električni bicikli i sl. (*„Standing/self balancing“*) i novijih modela električnih trotineta *(„Folding“*).

Cilj ovog rada jeste da detaljnije predstavi pogon električnog trotineta i da kroz simulacije odredi njegove ključne vozne osobine.



Sl. 1. Prikaz godišnje prodaje *LEV* vozila [2]

2. ELEKTRIČNA VOZILA MALE SNAGE

Električna vozila male snage ili *LEV* danas imaju veoma značajan procenat učešća u ukupnom broju električnih vozila u svetu. Njihova jednostavnost i pokretljivost čine ih idealnim za potrebe gradske vožnje. Postoji više vrsta ovih vozila, među kojima su električni trotineti, električni bicikli, automatizovani viljuškari, *hoverboard*-ovi, robotski transporteri i dr.

2.1 Električni bicikli

Električni bicikli su zapravo bicikli sa ugrađenim električnim motorom, baterijom i kontrolerom. Postoje dva podtipa električnih bicikala:

- sa pomoćnim pedalama (*Pedal Assist*), gde je električni motor regulisan okretanjem pedala i u slučaju detekcije

kretanja, senzor aktivira motor, kako bi olakšao kretanje vozača,

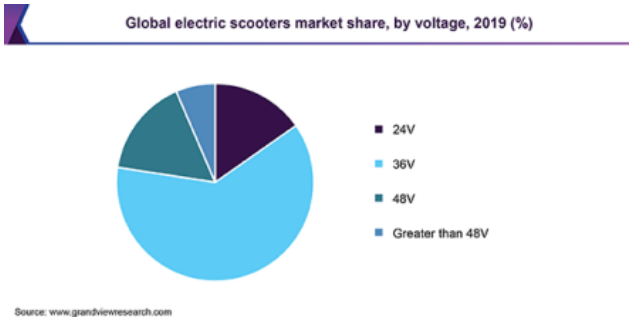
- sa pomoćnim pogonom (*Power on Demand*), gde se električni motor aktivira isključivo preko tastera, na zahtev korisnika i angažuje određena snaga motora.

Električni bicikli se prave ili kao zasebni tipovi ili kao kombinacija gore napomenutih modela.

2.2 Električni trotineti i *Hoverboard*-ovi

Električni trotineti su danas najpopularniji model električnih vozila male snage. Pogonski sklop električnog trotineta čini BLDC (*Brushless* DC) motor, koji se nalazi u samom točku trotineta i poznatiji je kao *hub* motor. Pogonski motori standardnih električnih trotineta su opsega od 300-500 W snage, dok naprednije verzije u sebi imaju ugrađene motore većih snaga i imaju neke druge namene, osim vožnje po asfaltu. Motor električnog trotineta može biti smešten u prednjem, zadnjem točku ili pak trotinet može imati dualne motore smeštene u oba točka [3]. Danas u svetu postoji mnogo različitih proizvođača električnih trotineta, a najistaknutiji su: *Xiaomi, Swagtron, Razor, Segway, Ninebot.*

Na slici 2 prikazani su naponi baterija električnih trotineta i njihovo učešće na tržištu. Primetno je da najveći deo električnih trotineta u sebi ima ugrađene baterije od 36 V, zatim slede 48 V, 24 V, dok se naponi većih od 48 V upotrebljavaju kod trotineta veće snage [4] .



Sl. 2. Naponski nivoi električnih trotineta [4]

Za razliku od električnih trotineta, *Hoverboard*-ovi imaju točkove paralelno postavljene jedan sa drugim, motori se nalaze u oba točka i kretanje se ostvaruje tako što vozač zauzima odgovarajući nagibni ugao. Žiroskopi detektuju promenu ugla i šalju motorima podatke, koji potom pogone *Hoverboard* određenom brzinom. Kompanija *Segway* poseduje patent na dizajn *Hoverboard* vozila sa stubom za lakše upravljanje. Zbog takvog dizajna, mnoge policijske ustanove širom sveta koriste upravo taj model *Segway Hoverboard*-a. Iako zamišljeni kao prevozno sredstvo, *Hoverboard* vozila uglavnom koriste mlađi naraštaji u svrhu razonode.

2.3 Robotski transporteri i automatizovani viljuškari

Automatizovani viljuškari su svoju svrhu pronašli u velikim magacinskim prostorima u kojima se brinu o transportu i skladištenju robe.

Robotski transporteri veliku primenu danas pronalaze u raznim granama privrede počevši od automobilske industrije do medicine. U automobilskoj industriji roboti služe za obavljanje manuelnih ponavljajućih funkcija gde su se znatno bolje pokazali od ljudskog faktora, dok se u medicini koriste kao ispomoć pri transportu medicinske opreme ili u samim procesima dezinfekcije potrebne aparature.

Neki od sistemi pomoću kojih se automatizovana vozila i roboti kreću u prostoru su: *Railed navigation, Wire based navigation, Lidar, Geo guidance*.

3. MODELOVANJE ELEKTRIČNOG TROTINETA

Modelovanje električnog trotineta i simulacija njegovog rada vršena je u programskom jeziku *Matlab* pomoću njegovog programskog paketa *Simulink*. Pri samoj simulaciji korišćena je verzija R2020a, jer u njoj postoji set blokova pod nazivom *Power Train Blockset*, koji je bio neophodan za modelovanje odgovarajućeg modela. Model trotineta, koji je modelovan je *Ninebot ES4*, čije specifikacije su prikazane u Tabeli 1.

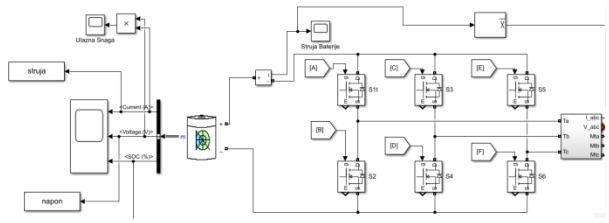
Tabela 1. Specifikacije električnog trotineta [5]

Model	Ninebot ES4
Maksimalna brzina (km/h)	30
Maksimalni domet (km)	44,8
Kapacitet Baterije (Wh)	374
Napon (V)	36
Ukupna masa (kg)	13,90
Vreme punjenja (h)	7
Nominalna snaga (W)	300
Maksimalna snaga (W)	500
Maksimalni uspon (°)	11
Kočnice	Mehaničke+električne
Trag zaustavljanja (m)	3,9624
Amortizeri	prednji +zadnji
Prednji točak - prečnik (inch)	8
Zadnji točak - prečnik (inch)	7.5
Maksimalna nosivost (kg)	100
Aplikacija/Bluetooth	да
LED Ekran	Mod, brzina, nivo bat
Sigurnost vidljivost	Prednji reflektor +zadnji i bočni
Dimenzije (V* Š *D)	(102*43*113)cm
Zaštita od vode	IP54
Preporučena starost	14-60 godina
Vrsta motora	BLDC motor

Model električnog trotineta sastoji se iz 4 podsistema koji su potom kompaktno uvezani u funkcionalnu celinu.

3.1 Energetski podsistem

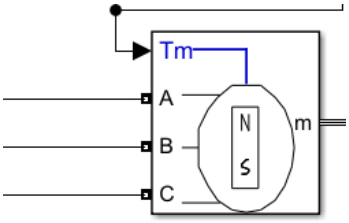
Energetski podsistem električnog trotineta sastoji se od li-Ion-ske baterije napona 36 V, kapaciteta 10,5 Ah. Kao inicijalni stepen popunjenosti baterije, uzeta je vrednost od 80%, dok su temperaturni efekti i efekti starenja baterije zanemareni. Na slici 3 prikazan je model energetskog podsistema, koji uključuje i trofazni invertor sačinjen od 6 parova MOSFET/dioda, koji omogućavaju dvosmerni tok energije.



Sl. 3. Energetski podsistem električnog trotineta

3.2 Pogonski podsistem

Pogonski podsistem sastoji se od odgovarajućeg trofaznog BLDC motora nominalne snage 300 W i maksimalne snage 500 W (Sl. 4). Povezivanjem motora sa *Bus selector* blokovima dobija se mogućnost očitavanja ugaone brzine, momenta, struja motora.



Sl. 4. BLDC Motor

Na ulaz T_m doveden je mehanički momenat, tačnije momenat opterećenja, koji se suprotstavlja samom kretanju vozila, a koji je izražen kao zbir sila otpora trenja i sile otpora vazduha. Sila otpora trenja je konstanta, dok se sila otpora vazduha menja u zavisnosti od trenutne brzine. Formule za proračun odgovarajućih sila koje se suprotstavljaju vozilu su date u nastavku. Sila otpora vazduha vozila i vozača je:

$$F_{trv} = 0{,}5\ \rho\ v^2\ C_w\ A \qquad (1)$$

gde su F_{trv} sila otpora vazduha u Njutnima; ρ gustina vazduha, ρ=1,18 kg/m^3; v brzina kretanja; C_w vrednost otpora vazduha, C_w=0,2; A frontalna površina vozila i vozača, A=0,75 m^2 [6]. Referentne brzine kretanja su zadate u km/h te je stoga potrebno izvršiti pretvaranja km/h u m/s. Sila otpora trenja je data sa:

$$F_{tr} = m\ g\ \mu \qquad (2)$$

gde su F_{tr} sila trenja u Njutnima; m masa vozila, m=85 kg; g ubrzanje gravitacija, g=9,81 m/s^2; μ koeficijent trenja, μ=0,002. Sada je ukupna sila, koja se suprotstavlja kretanju data sa:

$$F_{uk} = F_{tr} + F_{trv} \qquad (3)$$

a mehanički moment je dat sa:

$$T_m = F_{uk} \qquad (4)$$

Neke od vrednosti momenata, za odgovarajuće brzine kretanja, date su u Tabeli 2.

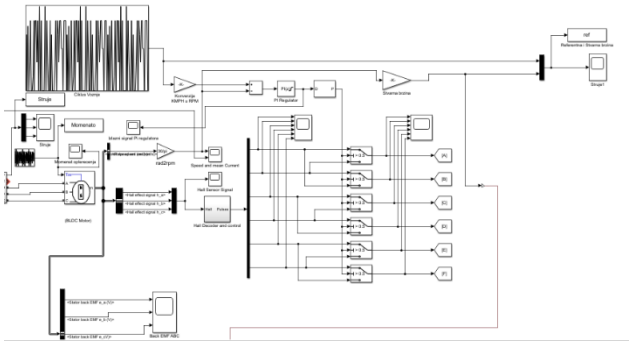
Tabela 2. Mehanički moment u zavisnosti od brzine

v [km/h]	5	10	15	20	25	30
T_m [Nm]	1.68	2.35	3.19	4.36	5.88	7.67

3.3 Upravljački podsistem

Na slici 5 prikazan je upravljački podsistem koji se sastoji od:

- bloka sa referencom brzine, tj. ciklusa vožnje u trajanju od 508 sekundi pomoću kojeg se zadaje željena brzina. Maksimalna brzina iznosi 30 km/h, jer su takve specifikacije električnog trotineta.
- PI regulator-a, koji poredi stvarnu i referentnu brzinu, te na taj način vrši upravljanje brzinom električnog trotineta. Vrednost proporcionalnog dejstva iznosi 15, dok vrednost integeralnog dejstva iznosi 20. Takođe, regulator ima podešenu opciju *anti wind-up*, kako ne bi ušao u ograničenje.

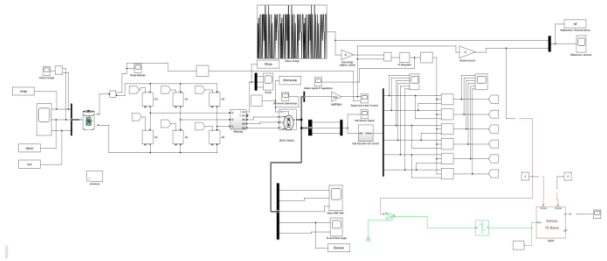


Sl. 5. Upravljački podsistem

Logika upravljanja poredi signale sa Hall dekodera i izlaza PI regulatora i na osnovu razlike parametara proračunava upravljačke signale za gejtove trofaznog invertora, koji se potom pomoću *Goto* blokova šalju na ulaze gejtova parova MOSFET/Dioda.

3.4 Mehanički podsistem

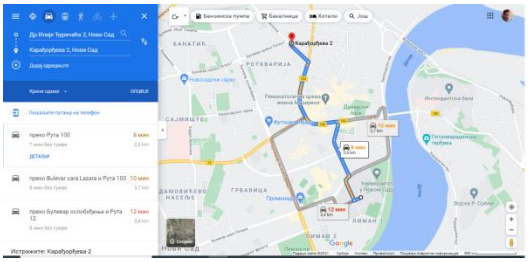
Mehanički podsistem sastoji se od samog tela vozila, bloka *Ideal Torque Source*, koji momenat na ulazu prosleđuje ka izlazu bez promene, menjača sa fiksnim prenosnim odnosom i osciloskopa za posmatranje izlaznih veličina. Masa vozila i vozača ukupno iznose 85 kg, pogonski momenat doveden je na prednji točak trotineta, a frontalni udari vetra i nagibni ugao su zanemareni. Kompletan model električnog trotineta dat je na slici 6.



Sl. 6. Model električnog trotineta

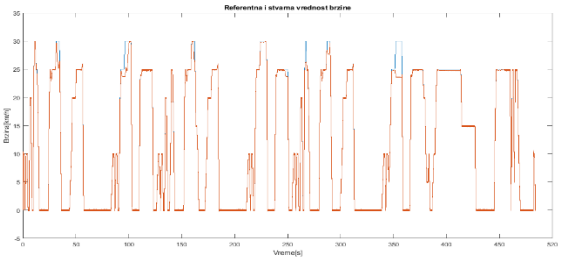
4. REZULTATI SIMULACIJE

Zamišljena putanja kretanja, kojom je simulirana trasa vožnje električnog trotineta, data je na slici 7. Predviđena je gradska vožnja po ulicama Novog Sada i to od lokacije u delu grada Podbara do cilja na Limanu I. Ciklus vožnje traje 508 sekundi i u sebi uvažava zaustavljanja na semaforima i raskrsnicama. Za vreme trajanja ciklusa vožnje električni trotinet prelazi rastojanje od 2,6 km i završava u neposrednoj blizini Fakulteta Tehničkih Nauka u Novom Sadu.



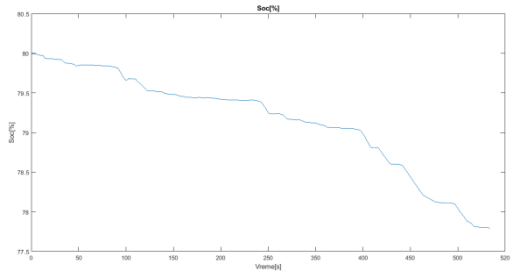
Sl. 7. Putanja kretanja zamišljenog ciklusa vožnje

Slika 8 prikazuje poređenje stvarne i referentne vrednosti brzine elekričnog trotineta izrežene u km/h. Sa grafika se može primetiti da električni trotinet dobro prati zadatu referencu brzine (referentna brzina označena je plavom dok je stvarna brzina označena crvenom bojom).



Sl. 8. Referentna i stvarna vrednost brzine

Na slici 9 prikazano je stanje napunjenosti baterije (*State of Charge*, SOC). Inicajlna popunjenost baterije iznosi 80% i po kretanju grafika mogu se primetiti nagla ubrzanja (pad SOC), ali i kratki periodi rekuperacije električne energije.



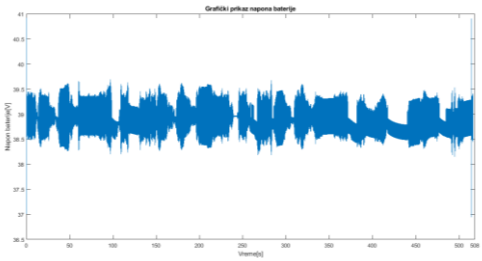
Sl. 9. Stanje SOC tokom ciklusa vožnje u [%]

Na slikama 10 i 11 prikazani su napon i struja baterije za vreme trajanja simulacije, respektivno. Može se uočiti da napona varira od 38,4 V do 39,7 V, dok struja od 0 do 12 A.

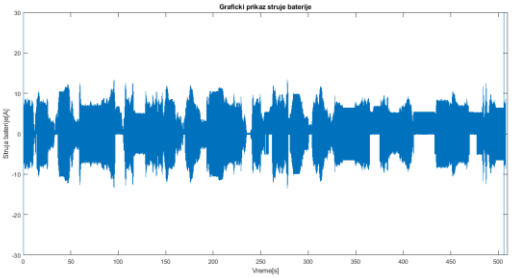
5. ZAKLJUČAK

Iz priložene simulacije može se primetiti da model u *Matlab-u* prilično verodostojno predstavlja stvarni električni trotinet. MBDS (*Model Based Developent System*) se pokazao kao koristan princip pri modelovanju vozila i njihovom testiranju pre nego što uđu u fazu proizvodnje. Iz rezultata simulacije može se zaključiti da su odzivi dobri i da se model ponaša veoma slično trotinetu u stvarnosti uz naravno idealizacije poput zanemarivanje frontalnog vetra i nagibnog ugla kretanja. Kroz rezultate je pokazano da se i električna energije rekuperiše.

Električni trotineti su najjednostavniji i najpogodniji za brzi transport u gradskim uslovima. Stoga u bliskoj budućnosti možemo očekivati još veći broj električnih trotineta na našim ulicama.



Sl. 10. Napon baterije



Sl. 11. Struja baterije

6. LITERATURA

[1] History of the Electric vehicle, https://en.wikipedia.org/wiki/History_of_the_electric_vehicle

[2] https://www.rutronik.com/suppliers/infineon/light-electric-vehicles/

[3] Segway_Inc, https://en.wikipedia.org/wiki/Segway_Inc.

[4] Electric scooters market, https://www.grandviewresearch.com/industry-analysis/electric-scooters-market

[5] Ninebot ES4, https://store.segway.com/segway-ninebot-kickscooter-es4

[6] How to calculate drag coefficient for motorcycle, https://info.simuleon.com/blog/how-to-calculate-drag-coefficient-for-motorcycle

Kratka biografija:

Filip Ačanski rođen je 1993. god. u Somboru. Osnovne studije završio je na Fakultetu tehničkih nauka 2019. god., a master 2021. god. iz oblasti Elektrotehnike i računarstva.

Vladimir A. 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 mu energetska elektronika, kvalitet električne energije, obnovljivi izvori električne energije i električna vozila.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.42:004.738.12
DOI: https://doi.org/10.24867/15BE13Dragutinovic

PREPOZNAVANJE PODATAKA O LIČNOSTI U TEKSTUALNIM DOKUMENTIMA

RECOGNITION OF PERSONAL DATA IN TEXTUAL DOCUMENTS

Đorđe Dragutinović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu prikazana je aplikacija za automatsko prepoznavanje podataka o ličnosti u tekstualnim dokumentima – specificirani su zahtevi i dizajn aplikacije, opisani su bitni elementi njene implementacije i demonstrirano je korišćenje aplikacije. Aplikacija je implementirana koristeći tehnike prepoznavanja imenovanih entiteta.*

Ključne reči: *imenovani entiteti; podaci o ličnosti; NER; SpaCy; Classla.*

Abstract – *This paper presents an application for automatic recognition of personal data in textual documents. The requirements and design of specified application are specified, the essential elements of its implementation are described and usage of the application is demonstrated. The application is implemented using named entity recognition techniques.*

Keywords: *named entities; personal data; NER; SpaCy; Classla.*

1. UVOD

Prema Zakonu o zaštiti podataka o ličnosti i Opštoj uredbi o zaštiti podataka (eng. *General Data Protection Regulation*, skr. GDPR), "podatak o ličnosti" je svaki podatak koji se odnosi na fizičko lice čiji je identitet određen ili odrediv, neposredno ili posredno, na osnovu oznake identiteta ili jednog, odnosno više obeležja njegovog identiteta. Neki od ličnih podataka koji se najčešće pominju i koriste su ime i prezime, adresa, jedinstveni matični broj građanina, broj telefona, ali u podatke o ličnosti spadaju i fotografija, otisak prsta, podaci o zdravstvenom i imovinskom stanju, verskim i političkim opredeljenjima i drugi [1, 2].

Pojava ovih podataka sve je češća u elektronskim dokumentima koji se čuvaju na računaru, a korisnici računara često nisu ni svesni da veliki broj dokumenata sadrži podatke o ličnosti.

Iz tog razloga, potrebno je rešiti problem prepoznavanja podataka o ličnosti u dokumentima, odnosno omogućiti analizu elektronskih dokumenata koji su sačuvani u memoriji računara i prepoznavanje podataka u ličnosti u tim dokumentima. Ovaj rad bavi se rešavanjem navedenog problema.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Stevan Gostojić, vanr. prof.

Navedeni problem najlakše je rešiti metodom koja se naziva prepoznavanje imenovanih entiteta *(eng Named Entity Recognition*, skr. NER). Ova metoda predstavlja jednu od tehnika obrade prirodnih jezika i spada u oblast veštačke inteligencije, a njen cilj je da u nestruktuiranom tekstu prepozna imenovane entitete i izvrši njihovu klasifikaciju, odnosno definiše kojoj kategoriji pripadaju. Termin „imenovani entitet“ odnosi se na sve što se može jednoznačno identifikovati, odnosno na sve što se može razlikovati od svih ostalih entiteta sa sličnim atributima.

Ostatak rada organizovan je na sledeći način: u drugom poglavlju analizirani su radovi koji su rešavali slične probleme. U trećem poglavlju specificirana je aplikacija koja je implementirana u okviru ovog rada. Četvrto poglavlje opisuje modele koji su iskorišćeni u implementaciji aplikacije. U poglavlju 5 prikazano je kako se opisano rešenje može iskoristiti za prepoznavanje imenovanih entiteta. U šestom poglavlju dat je zaključak o radu i navedene su njegove prednosti i mane, pri čemu je dat predlog kako se navedene mane mogu ispraviti.

2. STANJE U OBLASTI

Rešavanjem problema prepoznavanja imenovanih entiteta i podataka o ličnosti u tekstualnim dokumentima bavi se veliki broj postojećih radova. U radu [3], autori su prikazali kako se postojeći, javno dostupni modeli za prepoznavanje imenovanih entiteta mogu iskoristiti u analizi biografija. Skup podataka sastojao se od 249 ručno anotiranih biografija, preuzetih sa internet enciklopedije „Vikipedija“. Za prepoznavanje imenovanih entiteta iskorišćena su 4 postojeća, javno dostupna modela. Evaluacija korišćenih modela izvršena je prikazom preciznosti (eng. *precision*) i opoziva (eng. *recall*), ali su autori definisali i novi način evaluacije uspešnosti, kako bi se uzele u obzir i situacije kada je imenovani entitet fraza koja se sastoji iz više reči, a model uspešno prepozna samo jedan deo te fraze. Rezultati su pokazali da je preciznost prepoznavanja imenovanih entiteta varirala između 73% i 96%, a opoziv je varirao između 64% i 97%, u zavisnosti od korišćenog modela, a svi modeli su imali problem u prepoznavanju imenovanih entiteta koji se sastoje iz više reči.

Rad [4] bavi se poređenjem uspešnosti postojećih modela u prepoznavanju imenovanih entiteta u novinskim člancima napisanim na engleskom i ruskom jeziku. Korišćeno je ukupno 7 NER modela, a za njihovo testiranje iskorišćena su četiri javno dostupna skupa anotiranih novinskih članaka. Evaluacija korišćenih modela izvršena je prikazom F1 vrednosti (eng. *F1-score*), koja predstavlja harmoničku srednju vrednost preciznosti i opoziva. Rezultati su pokazali da su modeli mnogo preciznije prepoznavali

imenovane entitete u člancima napisanim na engleskom jeziku, a razlog za to leži u načinu obučavanja korišćenih modela.

Nekoliko radova bavilo se i problemom izdvajanja informacija iz dokumenata. Izdvajanje informacija podrazumeva da se mogu ručno definisati tipovi entiteta koje je potrebno prepoznati u dokumentima. U radu [5] prikazano je kako se može vršiti izdvajanje entiteta iz uputstava za upotrebu. Korišćeni skup podataka dobijen je ručnim anotiranjem 453 stranice različitih uputstava za upotrebu, napisanih na engleskom jeziku. Rezultati su pokazali da je preciznost prepoznavanja ručno definisanih kategorija entiteta varirala između 85% i 98%, što dovodi do zaključka da se NER modeli mogu koristiti i za izdvajanje informacija iz teksta.

3. SPECIFIKACIJA

Zadatak ovog rada je napraviti aplikaciju sa grafičkim korisničkim interfejsom, koja će korisnicima omogućiti da izvrše prepoznavanje imenovanih entiteta u tekstualnim dokumentima sačuvanim u memoriji računara. Poseban fokus potrebno je staviti na prepoznavanje podataka o ličnosti (pre svega, imena i prezimena osoba), kako bi korisnici imali uvid u to u kojim se dokumentima i u kom kontekstu pojavljuju ti podaci. Tipovi dokumenata koje je potrebno podržati su:

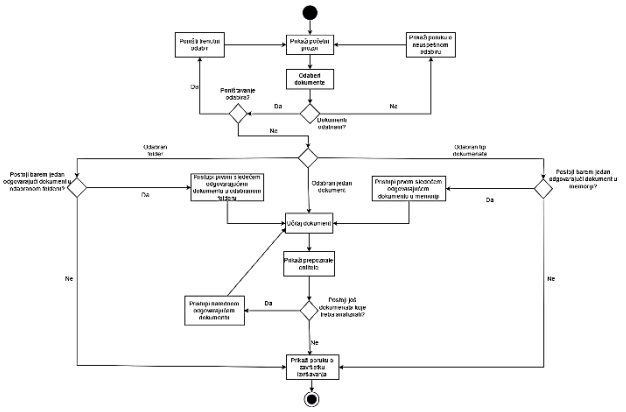
- *PDF*
- *txt*
- *doc(x)*

Samim tim, potrebno je korisnicima dati mogućnost da odaberu formate (jedan ili više) koje žele da uključe u pretragu. Takođe, osim mogućnosti odabira formata, korisnici bi trebalo da imaju i mogućnost da odaberu pojedinačni fajl ili folder u okviru kog će se vršiti analiza dokumenata. Prilikom odabira pojedinačnog fajla, korisnika treba ograničiti da može da bira samo dokumente odgovarajućeg formata. Ukoliko korisnik odabere folder u okviru kog će se vršiti analiza, izvršiće se rekurzivan pristup svim fajlovima i podfolderima unutar izabranog foldera.

Ukoliko korisnik pokuša da započne analizu dokumenata, a da prethodno nije odabrao format dokumenata koje želi da analizira ili pojedinačni dokument ili folder, korisniku je potrebno prikazati odgovarajuću poruku i vratiti ga na početak, kako bi mogao da pokuša ponovo. Ukoliko je korisnik odabrao pojedinačni fajl ili folder, potrebno mu je ponuditi mogućnost da poništi svoj odabir, ukoliko je pogrešio, te ga vratiti na početni ekran i omogućiti mu ponovno biranje ukoliko tu mogućnost iskoristi.

Prikazivanje prepoznatih imenovanih entiteta trebalo bi izvršiti tako što će se na glavnom prozoru aplikacije prikazati lokacija i kompletan sadržaj dokumenta koji se analizira u tom trenutku, pri čemu će imenovani entiteti biti prikazani drugačijom bojom u odnosu na ostatak teksta, a korisniku treba prikazati legendu, kako bi znao koji tipovi entiteta su obojeni i kojom bojom. Potrebno je obezbediti podršku za rad aplikacije sa dokumentima koji su napisani na srpskom ili engleskom jeziku, pri čemu je potrebno prilikom učitavanja dokumenta prepoznati na kom jeziku je dokument napisan, te u zavisnosti od toga iskoristiti odgovarajući NER model, takav da za prepoznati jezik daje najbolje moguće rezultate.

Dijagram aktivnosti specificirane aplikacije prikazan je na slici 1.



Slika 1. *Dijagram aktivnosti specificirane aplikacije*

4. IMPLEMENTACIJA

U implementaciji aplikacije za prepoznavanje imenovanih entiteta u tekstualnim dokumentima korišćeni su sledeći programski jezici, okruženja i biblioteke:

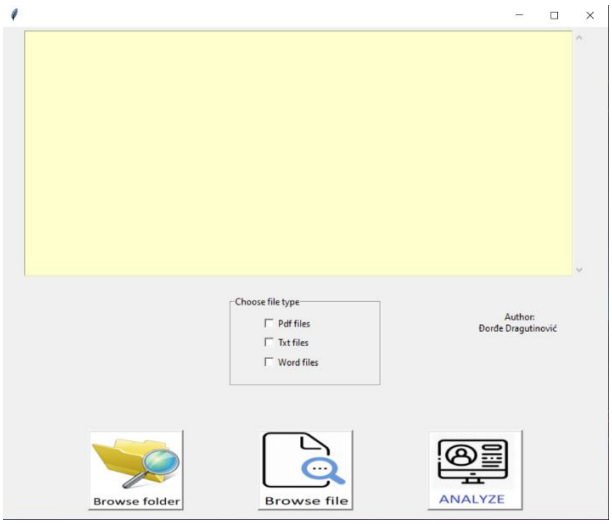
- *Python* – programski jezik visokog nivoa, opšte namene
- *PyCharm* – integrisano razvojno okruženje za programiranje u *Python*-u
- *Tkinter* – alat za kreiranje aplikacija sa grafičkim interfejsom u *Python*-u

Za prepoznavanje imenovanih entiteta u dokumentima iskorišćena su dva besplatna, javno dostupna modela:

- *SpaCy* – biblioteka otvorenog koda koja se koristi za naprednu obradu prirodnih jezika. Razvijena je od strane nemačke softverske kompanije „Explosion" i napisana je u programskom jeziku *Python*. Moguće je vršiti *alpha* tokenizaciju, *part-of-speech* tagovanje, kategorizaciju teksta, lematizaciju, segmentaciju rečenica, dodatno obučavanje modela i mnoge druge funkcije. U osnovi rada biblioteke nalaze se konvolutivne neuronske mreže i *Thinc* biblioteka za mašinsko učenje [6]. U implementiranoj aplikaciji, ovaj model iskorišćen je za prepoznavanje imenovanih entiteta u dokumentima napisanim na engleskom jeziku.
- Classla (*CLARIN Knowledge Centre for South Slavic languages*) – javno dostupni model za obradu i analizu tekstova napisanih na slovenskim jezicima (srpski, hrvatski, slovenački, makedonski, bugarski). Razvijen je od strane slovenačkog nacionalnog instituta CLARIN, a kao baza za kreiranje ovog modela iskorišćen je postojeći *Stanford NER* model, te se prepoznavanje imenovanih entiteta vrši kombinacijom HMM (eng. *Hidden Markov Model*) i MEMM (eng. *Maximum Entropy Markov Model*) algoritama. Korišćenjem ovog modela moguće je vršiti tokenizaciju teksta, *part-of-speech* tagovanje, lematizaciju, segmentaciju rečenica, parsiranje zavisnosti i prepoznavanje imenovanih entiteta [7]. U implementiranoj aplikaciji, ovaj model iskorišćen je za prepoznavanje imenovanih entiteta u dokumentima napisanim na engleskom jeziku.

5. DEMONSTRACIJA

Početni prozor implementirane aplikacije sastoji se iz 3 glavne celine. Prvu celinu predstavlja veliko polje (*textBox*) u okviru kog će se ispisivati rezultati operacija, kao i prepoznati entiteti. Ispod tog polja nalaze se 3 *checkButton* polja, pomoću kojih korisnik može odabrati koje tipove dokumenata želi da analizira (PDF, txt ili doc(x) dokumenti). Na dnu prozora nalaze se 3 dugmeta: "*browse folder*", koje nudi mogućnost odabira diska/foldera koji će se pretražiti, "*browse file*", koje nudi mogućnost odabira fajla koji će biti analiziran i "*analyze*", koje omogućava pokretanje procesa prepoznavanja entiteta. Početni prozor implementirane aplikacije prikazan je na slici 2.



Slika 2. *Početni prozor implementirane aplikacije*

Ukoliko je korisnik uspešno odabrao dokument ili folder koji želi da analizira, klikom na dugme "*analyze*" vrši se učitavanje sadržaja i prepoznavanje imenovanih entiteta. Na početku analize za svaki fajl će biti prikazana njegova lokacija, a taj tekst obojen je ljubičastom bojom.
Tekst analiziranog fajla biće prikazan u *textBox*-u u gornjoj polovini prozora, pri čemu će sve reči koje ne predstavljaju imenovane entitete biti obojene crnom bojom, a imenovani entiteti biće obojeni crvenom ili zelenom bojom, u zavisnosti od klase u koju su smešteni.
Entiteti koji predstavljaju imena i prezimena osoba, što su podaci o ličnosti koji se najčešće pojavljuju u dokumentima, biće obojeni crvenom bojom, dok će svi ostali imenovani entiteti (lokacije, organizacije, vremenske odrednice i drugi) biti obojeni zelenom bojom, a u donjem levom delu prozora pojavljuje se „oblačić" koji predstavlja legendu i korisniku pojašnjava način na koji su obojeni entiteti. Izgled prozora nakon prepoznavanja imenovanih entiteta u odabranom fajlu prikazan je na slici 3.

6. ZAKLJUČAK

U prethodnim poglavljima opisana je aplikacija koja korisnicima omogućava da izvrše prepoznavanje podataka o ličnosti u tekstualnim dokumentima sačuvanim u memoriji računara, napisanim na srpskom ili engleskom jeziku. Za implementaciju aplikacije iskorišćena je metoda koja se naziva prepoznavanje imenovanih entiteta (skr. NER).

Slika 3. *Prepoznavanje imenovanih entiteta u odabranom fajlu*

Aplikacija je kreirana korišćenjem programskog jezika *Python*, razvojnog okruženja *PyCharm* i biblioteke *Tkinter*.

Za prepoznavanje i klasifikaciju imenovanih entiteta iskorišćena su dva gotova modela: *SpaCy* i *Classla*. Najveće prednosti specificiranog i implementiranog rešenja jesu postojanje grafičkog interfejsa i mogućnost lakog odabira lokacije koju želimo da pretražujemo. Takođe, prednost implementiranog rešenja je i to što se korisnicima nudi mogućnost da odaberu tipove dokumenata koje žele da analiziraju.

Osim toga, dobra strana rešenja jeste način prikaza prepoznatih podataka o ličnosti u dokumentu. Umesto da bude naznačena samo pozicija podatka o ličnosti unutar teksta, prikazuje se kompletan tekst dokumenta, pri čemu su podaci o ličnosti posebno obojeni.

Najveća mana specificiranog i implementiranog rešenja jeste za nijansu slabija preciznost prepoznavanja imenovanih entiteta (izražena preko *precision* vrednosti) u odnosu na slična rešenja, analizirana u poglavlju 2. Još jedna mana implementiranog rešenja jeste nedostatak mogućnosti da se korisniku, ukoliko je odabrao folder ili disk u okviru kog će se vršiti analiza dokumenata, prikaže koliko se ukupno dokumenata (koje je moguće analizirati) nalazi u tom folderu/disku.

Ipak, navedene mane mogu se ispraviti kako bi aplikacija bila još učinkovitija, a modeli još precizniji u prepoznavanju imenovanih entiteta. Problem slabijih rezultata prepoznavanja entiteta može se rešiti tako što će se ručno izvršiti anotiranje velikog broja tekstova, dokumenata i novinskih članaka (posebno onih napisanih na srpskom jeziku), što će biti iskorišćeno za dodatno obučavanje modela.
Treba uzeti u obzir da bi trebalo anotirati tekstove iz što više različitih domena i oblasti, kako bi model „naučio" što više različitih reči, što bi omogućilo da ta znanja primeni za prepoznavanje entiteta u dokumentima koje korisnik odabere.
Mana koja se odnosi na prikaz broja dokumenata koji se nalaze u odabranom folderu mogla bi se rešiti tako što bi se, nakon odabira foldera, a pre pokretanja analize,

prebrojali svi dokumenti koji su sačuvani u odabranom folderu.

Taj broj bio bi prikazan korisniku, koji bi u tom slučaju mogao da proceni približno vreme izvršavanja analize i prepoznavanja imenovanih entiteta u dokumentima iz odabranog foldera.

7. LITERATURA

[1] Zakon o zaštiti podataka o ličnosti („Sl. glasnik RS“, broj 87/2018). [Online] Dostupno na: https://www.paragraf.rs/propisi/zakon_o_zastiti_podataka_o_licnosti.html. [Datum pristupa: 7. jul 2021.]

[2] Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) [Online] Dostupno na: https://gdpr-info.eu. [Datum pristupa: 7. jul 2021.]

[3] S. Atdag and V. Labatut, “A comparison of named entity recognition tools applied to biographical texts“, in Proc. of the 2nd International Conference on Systems and Computer Science, Villeneuve d'Ascq (FR), 2013 [Online]. Dostupno na: https://arxiv.org/abs/1308.0661. [Datum pristupa: 7. jul 2021.]

[4] S. Vychegzhanin and E. Kotelnikov, “Comparison of named entity recognition tools applied to news article”, in Proc. of the 2019 Ivannikov Ispras Open Conference (ISPRAS) [Online]. Dostupno na: https://ieeexplore.ieee.org/document/8991165. [Datum pristupa: 8. jul 2021.]

[5] C. M. Correia da Costa, G. Veiga, A. Jorge Sousa and S. Nunes, “Evaluation of Stanford NER for extraction of assembly information from instruction manual”, in Proc. of the 17th IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), April 2017 [Online]. Dostupno na: ieeexplore.ieee.org/document/7964092. [Datum pristupa: 9. jul 2021.]

[6] D. Altinok, *Mastering SpaCy: An end-to-end practical guide to implementing NLP applications using the Python ecosystem*, Packt Publishing LTD., Birmingham, UK, 2021.

[7] V. Batanović, N. Ljubešić, T. Samardžić and M. Miličević Petrović, “Otvoreni resursi I tehnologije za obradu srpskog jezika”, in Proc. of the Primena slobodnog softvera i otvorenog hardvera 2020 (PSSOH 2020), Beograd, Srbija [Online]. Dostupno na: www.researchgate.net/publication/349304650. [Datum pristupa: 13. jul 2021.]

Kratka biografija:

Đorđe Dragutinović rođen je 6.5.1997. god. u Zrenjaninu. Diplomski rad na Fakultetu tehničkih nauka odbranio je 2020. godine, a iste godine upisuje master studije na smeru Računarstvo i automatika – Inteligentni sistemi.

Kontakt: djordje.dragutinovic@uns.ac.rs

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.42:004.738.12
DOI: https://doi.org/10.24867/15BE14Trifkovic

KOMPARATIVNA ANALIZA OPEN-SOURCE ALATA ZA DIGITALNU FORENZIKU MOBILNIH UREĐAJA

COMPARATIVE ANALYSIS OF OPEN-SOURCE MOBILE DEVICE DIGITAL FORENSICS TOOLS

Bojan Trifković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTRONSKO POSLOVANJE

Kratak sadržaj – *Rad se bavi komparativnom analizom nekoliko besplatnih alata namenjenih za digitalnu forenziku mobilnih uređaja. Sadržaj je baziran na pojašnjenju osnovnih pojmova digitalne forenzike kao i opisa načina rada uz pomoć ovih alata.*

Ključne reči: *digitalna forenzika, mobilni uređaji, otvoreni kod, Deft, Santoku, Autopsy*

Abstract – *The paper deals with the presentation of open-source tools for digital forensics of mobile devices, with a part dedicated to digital forensics itself as a technological discipline. The paper describes how to use certain tools, as well as their advantages and disadvantages.*

Keywords: *digital forensics, mobile devices, , open-source, Deft, Santoku, Autopsy*

1. UVOD

Forenzika je nauka ili naučna disciplina koja se koristi u krivično pravnom sistemu [1].

Kompjuterska forenzika je posebna oblast forenzike, koja podrazumeva prikupljanje, analizu i ispitivanje podataka u cilju korišćenja istih kao dokaznog materijala protiv osumnjičenih ili u istragama [1].

Digitalna forenzika je nauka ili proces kroz koji se digitalni dokazi iz kompjutera, kamera, fotoaparata, memorijskih uređaja i slično, identifikuju, oporavljaju, čuvaju i prezentuju [1]. Postoji nekoliko vrsta softverskih alata i procesa koji se koriste za sprovođenje uspešne istrage. Među najpoznatije besplatne softverske alate spadaju *Autopsy*, *Deft* i *Santoku*.

Proces digitalne forenzike obezbeđuje istražiteljima uputstvo kako da upravljaju dokazima u toku istrage i kako da reprezentuju nalaze. Bez ovog uputstva dokazi mogu biti izgubljeni odnosno mogu se propustiti neke ključne stavke značajne za istragu [1].

2. FORENZIKA MOBILNIH UREĐAJA

Tokom 2018. godine više od polovine svih *online* aktivnosti pripisano je mobilnim uređajima i taj trend se i dalje nastavlja. Ovo je dovelo do toga da su oni kojima su potrebne razne informacije i cilju istrage ili obaveštajnih pitanja primorani da se obrate ekspertima za forenziku mobilnih uređaja.

2.1. Šta je forenzika mobilnih uređaja

Mobilna forenzika je vrsta digitalne forenzike koja ima za cilj preuzimanje podataka iz mobilnih uređaja, odnosno pametnih telefona i tableta. Mobilni uređaji poseduju razne tipove važnih podataka, počev od istorije poziva i SMS poruka do istorije internet pretraživača i istorije GPS lokacije, koja može dati informacije gde se vlasnik uređaja nalazio u određeno vreme [1].

O forenzici mobilnih uređaja najčešće se razmišlja u kontekstu sprovođenja zakona ali to ne mora da bude uvek tako, danas se i vojska često oslanja na informacije iz mobilnih uređaja prilikom planiranja svojih akcija ili protivterorističkih aktivnosti. Takođe i kompanije mogu ispitivati mobilne uređaje svojih zaposlenih ukoliko sumnjaju da im se krade intelektualno vlasništvo ili je zaposleni uključen u neke nedozvoljene aktivnosti.

2.2. Proces forenzike mobilnih uređaja

Ključ za prikupljanje digitalnih dokaza je poštovanje određenih pravila i praksi tokom tog procesa. Referentni model elektronskog otkrića nastao na Duke univerzitetu definisan je postupcima koji se koriste za prikupljanje elektronskih informacija.

2.3. Koraci u postupku ispitivanja digitalnih dokaza

Postupak ispitivanja digitalnih dokaza uključuje identifikaciju, pripremu, izolaciju, procesuiranje, verifikaciju, dokumentovanje, prezentovanje i čuvanje.

2.4. Mobilni i desktop uređaji u digitalnoj forenzici

Veoma često istražitelji moraju da rukuju istovremeno sa više digitalnih uređaja tokom istrage što znači da način rukovanja u velikoj meri može uticati na ishod istrage. Pored toga određeni operativni sistemi omogućavaju mobilnim i desktop uređajima da lako razmenjuju informacije između dve vrste uređaja. Na primer *Apple MacOS* i *iOS* rade u tandemu kako bi omogućili korisnicima da rad sa desktop uređaja mogu lako da nastave prebacivanjem na mobilni uređaj i obrnuto. Ova mogućnost nazvana "kontinuitet" je primer kako odnos između mobilnih i desktop uređaja može uticati na forenzičarske istrage, što znači da istražitelji mogu da rade na jednom uređaju i lako se po potrebi prebace na drugi. Ono što digitalni forenzičari moraju uvek da pretpostave jeste da napadi mogu biti izvršeni korišćenjem više uređaja od strane jednog hakera ili grupe.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Stevan Gostojić, vanr. prof.

2.5. Operativni sistemi mobilnih uređaja

Mobilni operativni sistem je operativni sistem namenjen za mobilne telefone, tablete, pametne satove i druge mobilne uređaje. Putem operativnog sistema mobilni uređaj upravlja svojom memorijom i resursima. Postoji više vrsta operativnih sistema mobilnih uređaja, neki od najpoznatijih operativnih sistema za pametne mobilne uređaje su *Anrodid* i *iOS*. Mobilni uređaji sadrže mnoštvo interfejsa ka spoljašnjem svetu.

Bezbednost ovih interfejsa od različitih napada i neovlašćenog korišćenja je ključna stavka bezbednosti mobilnih telefona danas [1].

Mobilni telefoni imaju mnogo interakcija sa spoljašnjim svetom: Wi-Fi, Bluetooth, SD kartice, USB portovi itd. Takođe, većina korisnika ima pristup bankovnim računima, platnim sistemima banke, poverljivim dokumentima, lozinkama i sličnom u svom telefonu. Ovo ima velike rizike po korisnike telefona.

U cilju zaštite mobilnih uređaja koriste se različiti mehanizmi autentifikacije. Neki od najvećih rizika za mobilne uređaje su: curenje podataka, mešanje poslovnog i privatnog sadržaja i napadi koji su usmereni na ubacivanje *malware* softvera.

2.6. Uobičajeni tipovi podataka u forenzici mobilnih uređaja

Mobilni telefoni sadrže različite tipove podataka koji se mogu prikupiti i analizirati u forenzičkoj istrazi. Najčešće se obraća pažnja na nekoliko mesta na kojima se čuva velika većina podataka, a to su memorija telefona, SIM kartica kao i spoljni uređaji za čuvanje podataka. Digitalni forenzičari se najčešće bave podacima koji se nalaze u memoriji uređaja. Uobičajeni tipovi podataka povezanih sa mobilnim uređajima koji se koriste u istragama su: istorija poziva, kontakti, *SMS* i *MMS* poruke, multimedijalni sadržaj, elektronska pošta, istorija internet pretraživača, podaci aplikacija, istorija lokacije, kalendar, obrisani podaci itd.

Nekim od navedenih podataka se može lako pristupiti (npr. fotografijama, porukama, elektronskoj pošti). Ostalim podacima posebno onim pohranjenim u aplikacijama, može biti teže pristupiti ili mogu biti šifrirani.

2.7. Prikupljanje i analiza dokaza u forenzici mobilnih uređaja

Od istražitelja se očekuje poznavanje industrijskih standarda i najboljih praksi prikupljanja i analize dokaza tako da se prikupljeni podaci mogu iskoristiti i čuvati na sudu. Da bi se dokazi iz mobilnih uređaja mogli proći kontrolu na sudu, postoje određena pravila koja bi dokazi trebalo da ispoštuju: autentičnost, pouzdanost, verodostojnost, temeljnost, prihvatljivost.

Postoje najčešće tri vrste prikupljanja podataka sa mobilnih uređaja: ručno, logičko, fizičko. Najkritičniji aspekt prikupljanja podataka je održavanje integriteta dokaza. Bez obzira koja metoda za pribavljanje se koristi, dokumentovanje tog procesa je ključno za stvaranje dokaza koji su verodostojni i pouzdani.

2.8. Klasifikacioni sistem za forenziku mobilnih uređaja

Klasifikacioni sistem kreiran je sa ciljem kako bi istražiteljima dao pregled dostupnih alata koji se koriste u svrhu prikupljanja dokaza kao i njihov način upotrebe, počev od najmanje kompleksnih pa do onih najsloženijih.

Postoji pet nivoa tehnika za izvlačenje podataka: manuelno, logičko, *hex dump, chip off* i mikro čitanje.

3. ALATI ZA DIGITALNU FORENZIKU MOBILNIH UREĐAJA

Alati za digitalnu forenziku dele se u više kategorija, tako da tačan izbor alata zavisi od potreba i mogućnosti korisnika. Postoji nekoliko kategorija kojima su namenjeni alati: forenzika baza podataka, analiza elektronske pošte, forenzika audio i video sadržaja, analiza internet pretrage, forenzika mreže, forenzika operativne memorije uređaja, analiza fajl sistema, kompjuterska forenzika.

S obzirom na mnoge mogućnosti, nije lako odabrati pravi alat koji će odgovarati potrebama istražitelja. Neki od aspekata koje treba razmotriti tokom donošenja odluke su: nivo veština, izlazni rezultati, budžetska ograničenja, područje fokusa istrage i dodatna oprema.

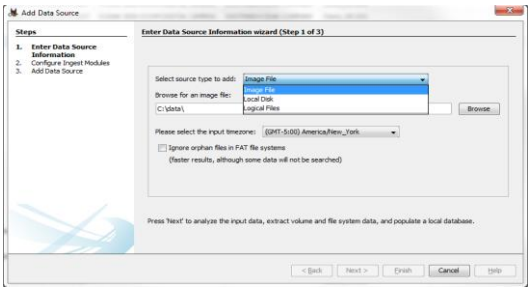
Neki od najpopularnijih besplatnih alata za digitalnu forenziku su [2]: *The Sleuth Kit and Autopsy, SIFT, Deft, Santoku* i *Caine.*

3.1. Autopsy i Sleuth Kit

Autopsy je softverski alat, jednostavan za upotrebu, zasnovan na *GUI* – u, koji omogućava efikasnu analizu diskova i pametnih telefona. Radi na *Windows* operativnom sistemu.

Sleuth Kit je kolekcija command line alata i biblioteka u programskom jeziku C koji omogućavaju analizu diska i vraćanje obrisanih datoteka. Koristi se iza scene u *Autopsy* alatu i mnogim drugim open source i komercijalnim alatima za forenziku.

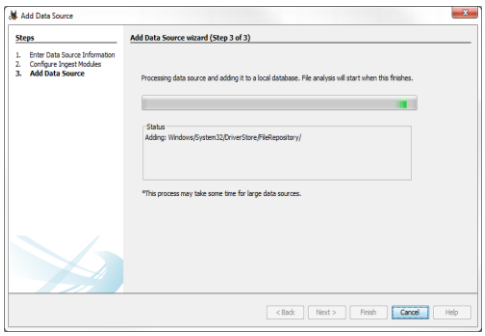
Osnovni koncepti Autopsy alata su: izvor podataka (*data source*), slučaj (*case*) i centralna baza podataka. Na slici 1 prikazan je dijalog prozor gde korisnik bira tip izvora podataka.



Slika 1. *Dijalog prozor za dodavanje izvora podataka*

Na slici 2 prikazan je dijalog prozor gde korisnik prati status osnovnog ispitivanja izvora podataka.

Pre dodavanja izvora podataka korisnik mora da kreira novi slučaj. Autopsy podržava tri vrste izvora podataka: slika diska, lokalni disk i logičke datoteke.



Slika 2. *Dijalog prozor statusa ispitivanja izvora podataka*

Autopsy je softverski alat koji omogućava veoma efikasnu i pouzdanu analizu čvrstih diskova kao i pametnih telefona, tableta i drugih *Androd* uređaja. Karakteriše ga korisnički interfejs koji je veoma intuitivan i lak za upotrebu, brzo procesuiranja podataka i niski troškovi. Sleuth Kit je kolekcija sastavljena od command line alata i biblioteka u programskom jeziku C koji omogućavaju analizu slike diska i oporavak fajlova i podataka. To se zapravo koristi kao pozadina Autopsy alata, odnosno kao njegov backend alat.

3.2. Deft

Deft je distribucija *Linux* operativnog sistema namenjena za digitalnu forenziku. *Deft* se smatra najboljim izborom među bezbedonosnim agencijama i agencijama za sprovođenje zakona za forenzičarske istrage [3]. Slika 3 prikazuje korisnički interfejs ovog operativnog sistema.

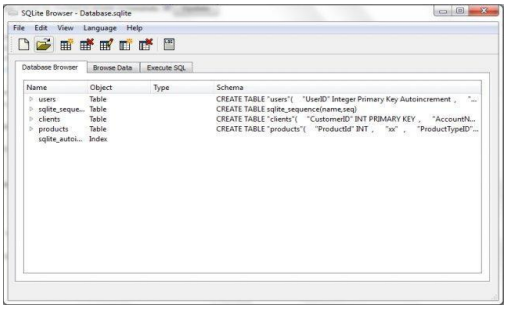
Slika 3. *Izgled Deft operativnog sistema*

Deft sadrži *DART* paket koji sadrži *Windows* aplikacije koje su i dalje održive jer u *Unix* operativnim sistemima ne postoji ekvivalent. Pokretanje *Windows* aplikacija na *Linux* operativnom sistemu u slučaju *Deft* distribucije moguće je uz pomoć softverskog alata koji se naziva *Wine*. *DART* je aplikacija koja organizuje, prikuplja i pokreće softver u sigurnom režimu u svrhu forenzičke analize uživo i reagovanja na bezbednosne incidente.

Deft Linux uključuje i neke alate za analizu mobilnih uređaja. Dostupan je pretraživač *SQLite* baze podataka koji omogućava analizu podataka koji se koriste u većini aplikacija za *Android*, *iPhone* i *iPad* uređaja.

Postoji i *Ipddump* za analizu backup-ova *BlackBerry* uređaja, kao i *iPhone analyzer* za analizu *iPhone* uređaja koji je dostupan od verzije 3 pa na više [4]. S obzirom da *Android* i *iPhone* pametni telefoni većinu podataka čuvaju upravo u *SQLite* bazi podataka, *SQLite database browser* omogućava korisniku uvid u podatke kroz grafički prikaz.

Na slici 4 prikazan korisnički interfejs taba *Database browser* u okviru *SQLite database browser* alata.



Slika 4. *Database browser* tab u okviru *SQLite database browser* alata

Za pretragu fajlova i direktorijuma u okviru *Deft* operativnog sistema može se koristiti softverski alat *Catfish*. *Catfish* može izvoditi iste operacije koje se mogu izvršiti putem naredbi komandne linije *find* i *locate*.

Prednosti *Deft*-a su jako mali memorijski zahtevi za pokretanje aplikacije (svega 400 MB). Ovo znači da može biti pokrenut i na starijim, sporijim računarima.

3.3. Santoku

Santoku Linux je besplatni projekat otvorenog koda sponzorisan od strane kompanije "*NowSecure*" iz Čikaga, čiji članovi čine jezgro razvojnog tima [5]. *Santoku* je napravljen sa ciljem da omogući rad u polju mobilne forenzike i analize bezbednosti i upakovan je u platformu otvorenog koda koja se lako koristi. *Santoku* uključuje veliki broj alata otvorenog koda koji omogućavaju rad sa svim aspektima mobilne forenzike, analize *malware*-a i bezbednosno testiranje. Izgled UI Santoku operativnog sistema prikazan je na slici 5.

Slika 5. *Izgled Santoku operativnog sistema*

Tipovi forenzike podržani od strane *Sandoku* operativnog sistema i njegovih alata su: logička, forenzika fajl sistema i forenzika fizičkih diskova. Logička analiza i pribavljanje podataka vrši se uz pomoć aplikacije *AF Logical OSE* iz menija *Device Forensics*. Rezultat ove analize može dati uvid u razne vrste podataka na telefonu kao što su kontakti, slike, istorija poziva i SMS poruke. Prednosti *Santoku* operativnog sistema su sto je brz za ovu vrstu analize. Forenzika fajl sistema omogućava analizu veće količine podataka nego logička analiza, ali zahteva i dodatni pristup uređaju. Fizička analiza podrazumeva kopiranje bit po bit fizičkog memorijskog uređaja i to je najčešće korišćena forenzička tehnika iz razloga što su najveće šanse za oporavak obrisanih podataka [6].

4. DISKUSIJA

Ovaj rad se bavi analizom tri besplatna alata namenjenih digitalnoj forenzici mobilnih uređaja. Stiče se utisak da razvoj besplatnih alata za forenziku mobilnih uređaja još uvek nije dosegao nivo koji bi omogućio veću zainteresovanost inženjera za njihovo istraživanje, upotrebu ili direktno učestvovanje u razvoju istih.

Autopsy se može smatrati alatom koji je najsličniji komercijalnim alatima za digitalnu forenziku. Razlog ovog mišljenja jeste način upotrebe ovog alata i njegov napredni korisnički interfejs koji omogućava kreiranje vizuelnih i lako čitljivih izveštaja o rezultatima operacija. Za razliku od *Deft* i *Santoku* operativnih sistema koji pored mogućnosti obavljanja operacija digitalne forenzike imaju još dodatnih funkcionalnosti, *Autopsy* je softver primarno namenjen digitalnoj forenzici.

Deft je verzija *Linux* operativnog sistema bazirana na Ubuntu distribuciji koja nudi preko 1GB dodatnog besplatnog softvera. *Deft* poseduje mogućnost pokretanja *Windows* aplikacija uz pomoć alata koji se zovu *LXDE* i *WINE*, dok *Santoku* nema mogućnost rada sa ova dva softvera jer nije baziran na *Ubuntu* distribuciji operativnog sistema.

Deft Linux takođe nudi mogućnost obavljanja forenzičarskih zadataka iz domena pametnih i mobilnih uređaja. *SQLite database browser* omogućava pregled baza podataka *Android* aplikacija u okviru telefona. Nudi mogućnost tabelarnog prikaza kroz intuitivan korisnički interfejs. Korisnički interfejs na visokom nivou omogućava istražiteljima jasan uvid u podatke u okviru aplikacije, što znatno ubrzava i olakšava istrage.

Santoku je *Linux* distribucija čiji je glavni fokus na mobilnoj forenzici, a takođe uključuje alate koji omogućavaju brute force dešifrovanje *Android* uređaja, analiza backup-a *iPphone* uređaja kao i automatsko prepoznavanje priključenih uređaja.

Santoku nudi mogućnost obavljanja tri tipa analize fajlova – logička analiza, forenzika fajl sistema i forenzika fizičkih diskova. Fizička analiza je najpreciznija i najčešće korišćena tehnika analize memorijskih uređaja uz pomoć *Santoku* operativnog sistema, jer podrazumeva kopiranje bit po bit sadržaja memorijskog uređaja te samim tim obezbeđuje velike šanse da se dođe do svih neophodnih podataka. Logička analiza je proces koji takođe iziskuje određeno iskustvo u radu sa *Linux* operativnim sistemom, jer podrazumeva izvršavanje *Linux* komandi u terminalu u toku procesa analize.

Analiza i presretanje mrežnog saobraćaja uz pomoć Santoku Linux operativnog sistema je izuzetno efikasna i uspešna operacija. Od istražitelja koji obavljaju ove zadatke očekuje se određeni nivo predznanja pre svega o mrežnom saobraćaju i *HTTP* protokolu.

Santoku i *Deft* je moguće pokrenuti sa *Windows* operativnog sistema uz pomoć *VMware* i *Virtualbox* softvera (virtuelne mašine). *Autopsy* ne zahteva instalaciju dodatnog operativnog sistema i namenjen je radu na *Windows* operativnom sistemu.

5. ZAKLJUČAK

Digitalna forenzika je oblast koja igra veoma značajnu ulogu u informacionim društvu. Zbog sve veće količine podataka pretpostavke su da će digitalna forenzika imati veoma zapaženu ulogu u budućnosti, jer sa porastom količine podataka raste i rizik od njihove krađe i zloupotrebe.

Trenutno, analizi za digitalnu forenziku mobilnih uređaja otvorenog koda po funkcijama koje nude zaostaju za vlasničkim alatima.

Poslednjih godina primećen je pomak u razvoju alata otvoren koda ka usvajanju i uvođenju novih funkcionalnosti što ih čini pogodnijim i konkurentnijim u budućem radu i korišćenju.

6. BIBLIOGRAFIJA

[1] Computer Science: Mobile Forensics, https://study.com/academy/course/computer-science-335-mobile-forensics.html (pristupljeno u julu 2020.)

[2] Autopsy User Documentation 4.3, https://sleuthkit.org/autopsy/docs/user-docs/4.3/ (pristupljeno u oktobru 2020.)

[3] Forensic Investigation Tutorial Using DEFT, https://www.hackingarticles.in/forensic-investigation-tutorial-using-deft/ (pristupljeno u oktobru 2020.)

[4] DEFT Linux A Linux Distribution For Computer Forensics, http://www.linuxandubuntu.com/home/deft-linux-a-linux-distribution-for-computer-forensics (pristupljeno u martu 2021.)

[5] How to use Santoku in Andorid Forensics?, https://infosecaddicts.com/use-santoku-android-forensics/ (pristupljeno u martu 2021.)

[6] About Santoku, https://santoku-linux.com/about-santoku/ (pristupljeno u aprilu 2021.)

Kratka biografija:

Bojan Trifković rođen je u Petrovu, BiH 1992. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Komparativna analiza open-source alata za digitalnu forenziku odbranio je 2021.god.
kontakt: bojantrifkovic92@gmail.com

Зборник радова Факултета техничких наука, Нови Сад

UDK: 007:004.056
DOI: https://doi.org/10.24867/15BE15Matijevic

ВЕБ АПЛИКАЦИЈА ЗА ВИЗУАЛИЗАЦИЈУ TLS ПРОТОКОЛА

WEB APPLICATION FOR TLS PROTOCOL VISUALIZATION

Милица Матијевић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *У раду је анализиран механизам функционисања различитих верзија криптографског протокола TLS – SSL 3.0, TLS 1.0, TLS 1.1, TLS 1.2 и TLS 1.3. Размотрени су различити приступи учењу и разумевању принципа на којима се заснива TLS протокол, као што су алат JCrypTool [1] и вебсајт за визуализацију TLS протокола [2]. Напослетку је дат опис Веб апликације за визуализацију TLS протокола.*

Кључне речи: *криптографија, криптографски протокол, SSL, TLS, handshake протокол*

Abstract – *This paper analyzes the TLS protocol versions such as SSL 3.0, TLS 1.0, TLS 1.1, TLS 1.2, and TLS 1.3. It gives an overview of the TLS protocol learning approaches which representatives are JCrypTool [1] and website for TLS protocol visualization [2]. Finally, the Web application for TLS protocol visualization is described.*

Keywords: *cryptography, cryptographic protocol, SSL, TLS, handshake protocol*

1. УВОД

Комуникација на даљину данас је свеприсутна захваљујући добро успостављеном механизму функционисања комуникационих мрежа, као и комуникационих протокола. Предмет дискусије у вези са овим видом комуникације често су поверљивост, поузданост и безбедност. С једне стране, поставља се питање у којој мери је неопходно постизање приватности, односно тајности, у комуникацији на даљину, а с друге стране, изражава се забринутост због неадекватних мера заштите информација које се преносе, или пак због њиховог потпуног изостанка. Да би се дискусија уопште водила, потребно је разумевање безбедносних комуникационих протокола. Криптографија, као наука на којој се поменути безбедносни протоколи заснивају, изучава се у оквиру појединих инжењерских дисциплина, међутим, како су питања тајности комуникације на даљину данас постала и део правног дискурса, то се важност разумевања безбедносних механизама рачунарске мреже, проширује. С тим у вези, овај рад даје преглед *SSL/TLS* криптографског протокола и опис Веб апликације за визуализацију *TLS* протокола имајући за циљ олакшавање разумевања принципа на којима се поменути протокол заснива.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Горан Сладић, ванредни професор.

2. *SSL/TLS* ПРОТОКОЛ

SSL/TLS криптографски протокол, у оквиру *World Wide Web (WWW)* платформе, омогућава безбедну комуникацију у рачунарској мрежи клијент-сервер архитектуре. Безбедносни принципи, које овај протокол обезбеђује, су међусобна аутентификација страна које комуницирају, аутентификација порука, поузданост скупа података који се размењују у току сесије повезаности, као и интегритет скупа података без могућности опоравка у случају губитка података. *SSL/TLS* протокол не обезбеђује непорецивост података [3].

У оквиру *OSI* референтног модела, *SSL/TLS* протокол се састоји од два поднивоа са потпротоколима. Доњи подниво, *Record* протокол, ослања се на протокол нижег нивоа, што је у случају *TCP/IP* комуникационог модела – *TCP* протокол. Горњи подниво састоји се од четири потпротокола: *Handshake*, *ChangeCipherSpec*, *Alert* и *Application* протокол. *Handshake* протокол (протокол руковања) је основни *SSL/TLS* потпротокол и служи за аутентификовање страна које комуницирају, као и за обезбеђивање интегритета и поузданости размењених података. Свака порука протокола руковања започиње пољем са вредношћу ознаке типа поруке. Затим следе поља са вредностима ознаке верзије *SSL/TLS* протокола и дужине поруке. Поруке које се размењују у оквиру протокола руковања између клијента и сервера су: *ClientHello*, *ServerHello*, *Certificate*, *ServerKeyExchange*, *CertificateRequest*, *ServerHelloDone*, *Certificate*, *ClientKeyExchange*, *CertificateVerify*, *ChangeCipherSpec*, *Finished*, *ChangeCipherSpec* и *Finished [4]*.

SSL/TLS протокол је блок-оријентисан протокол, величине блока један бајт, те свако поље *SSL/TLS* поруке представља мултипликацију једнобајтног блока [4].

Безбедносни механизми *SSL/TLS* протокола заснивају се на криптографским техникама, те се за аутентификацију порука и шифровање података користи криптографија тајног кључа, док се за аутентификацију страна које комуницирају, као и за успостављање кључа шифровања, користи криптографија јавног кључа. У основи постоје три алгоритма која се користе за успостављање вредности кључа: *RSA*, Дифи-Хелман (енг. *Diffie-Hellman)* и *FORTEZZA*. Најсигурнија верзија Дифи-Хелман алгоритма користи краткотрајне вредности кључа које се генеришу са сваком новом применом протокола руковања (енг. h*andshake protocol*).

Оваква имплементација пружа заштиту од ретроактивног компромитовања кључа сесије ако је већ дошло до компромитовања дуготрајног материјала за генерисање кључева (енг. *perfect forward secrecy*) [3].

3. SSL 3.0, TLS 1.0, TLS 1.1, TLS 1.2 и TLS 1.3

Евидентнија разлика између SSL 3.0 и TLS 1.0 протокола везује се за начин креирања кључева, где важну улогу има функција за генерисање случајних вредности (енг. *Pseudorandom Function, PRF*). TLS 1.0 протокол у те сврхе користи функцију која се базира на функцији експанзије података, а која се заснива на криптографској хеш функцији (MD5 или SHA-1).

Због великог броја напада који су реализовани над рањивостима TLS протокола верзије 1.0, верзија 1.1 садржи бројне модификације. Између осталог, промена се огледа у скупу комбинација криптографских техника који не садржи комбинацију која укључује ланчање блок-шифре (*cipher block chaining, CBC mode*) [3].

Верзија 1.2 TLS протокола објављена је 2008. године у оквиру публикације RFC 5246 (*Request for Comments*). Дистинкција између протокола TLS 1.2 и TLS 1.1 огледа се у много чему. Итеративни поступак генерисања псеудослучајних бројева (енг. *PRF*), као и поступак дигиталног потписивања у оквиру TLS 1.2 протокола искључују комбиновање двеју хеш функција (MD5 и SHA-1), већ користе само једну од њих [3] [5]. Такође, TLS 1.2 протокол одликује се проширењима у виду додатних поља порука *ClientHello* и *ServerHello* као и нових типова порука, које ова проширења захтевају. Проширења омогућују флексибилнију негоцијацију криптографских техника. При томе је комуникација између клијента и сервера могућа и ако један од њих не подржава поменута проширења [3].

Успостављање TLS конекције прописано 1.3 верзијом протокола, захтева размену порука између клијента и сервера у свега три наврата, те се може упоредити са успостављањем TCP конекције. Клијент најпре шаље *ClientHello* поруку, за којом одмах следи *ClientKeyShare* порука. Порука *ClientKeyShare* представља замену за *ClientKeyExchange* поруку, присутну у претходним верзијама TLS протокола, а коју карактерише размена статичких кључева. Ради веће безбедности, размена статичког материјала за кључеве је у 1.3 верзији искључена, те у обзир долазе само привремени кључеви (енг. *ephemeral*) који обезбеђују механизам заштите од малициозног ретроактивног дешифровања порука (енг. *perfect forward secrecy*) [3].

4. ПРЕГЛЕД ПОСТОЈЕЋИХ РЕШЕЊА

У наставку су дата постојећа решења у пољу криптографије и безбедносних алгоритама и протокола са едукативног аспекта.

4.1. JCrypTool

JCrypTool је библиотека која имплементира криптографске механизме, те тако омогућава дизајн и симулацију криптографских система. Истоимени алат који се заснива на овој библиотеци, нуди графички интерфејс путем кога је омогућена интеракција са корисником. Између осталог, *JCrypTool* алат омогућава и симулацију *SSL/TLS* протокола уз увид у све параметре и њихову поставку у складу са стандардима [1].

4.2. Имплементација и симулација *SSL* протокола у *WPF* технологији

Допринос скупу едукативних алата у области криптографије, дао је и Харш Вахарајани (Harsh Vachharajani) у својој мастер тези под оригиналним називом *Implementation and Simulation of Secure Sockets Layer (SSL) in Windows Presentation Foundation [6]*. Имплементација *SSL* протокола у овоме раду реализована је помоћу библиотеке *Mentalis*, која је отвореног кода и креирана је за *.NET* радни оквир, а садржи модуле са имплементираним криптографским механизмима протокола *SSL 3.0* и *TLS 1.0*.

Графички кориснички интерфејс алата описаног овом тезом, имплементиран је употребом *WPF* технологије, а поред демонстрационог режима, алат нуди и режим за симулацију напада на механизме функционисања *SSL* протокола.

4.3. Илустрована *TLS* конекција

Међу великим бројем веб страница које нуде објашњења *SSL/TLS* протокола, издваја се илустрација [2], која обухвата детаљан увид у све поруке које се размене током протокола руковања.

Ниво детаљности огледа се у бајтном запису порука, те анотацијама које дају објашњење сваке елементарне групе бајтова.

5. ВЕБ АПЛИКАЦИЈА ЗА ВИЗУАЛИЗАЦИЈУ *TLS* ПРОТОКОЛА

У овом поглављу описана је Веб апликација за визуализацију *TLS* протокола верзије 1.2, при чему је дат модел решења и важнији имплементациони детаљи. Радни оквир коришћен за имплементацију веб апликације је *Flask*.

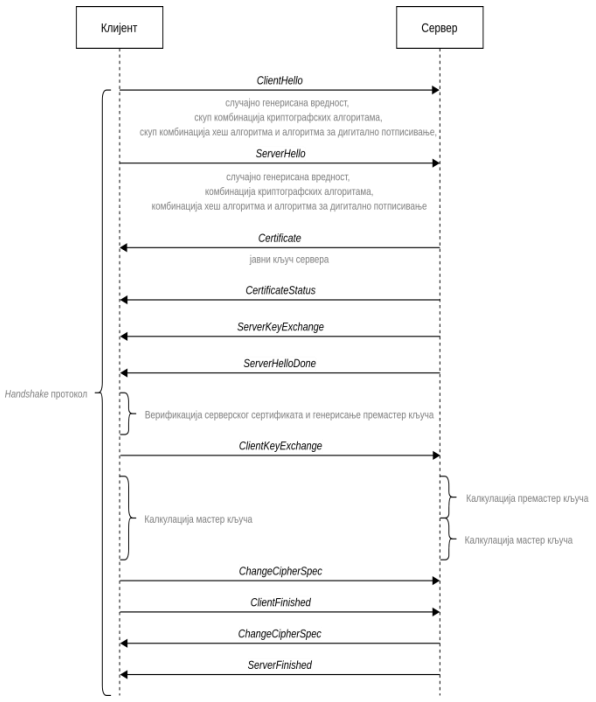
5.1. Модел решења

Дијаграмом секвенце са Слике 1 представљен је ток размене порука које подржава Веб апликација за визуализацију *TLS* протокола.

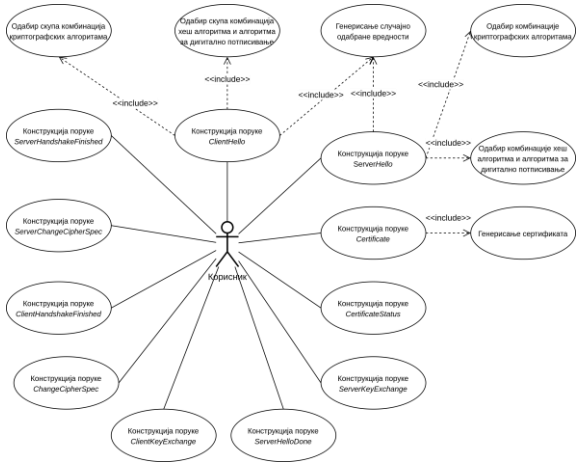
Такође, у виду коментара одговарајуће поруке, приказане су вредности чија је калкулација повезана са датом поруком.

На Слици 2 дат је дијаграм случајева коришћења Веб апликације за визуализацију *TLS* протокола.

Корисник је у могућности да иницира симулацију сваке од порука које се размењују у оквиру протокола руковања, као и да иницира калкулације као што су генерисање материјала за кључеве, генерисање сертификата итд.



Слика 1: *Дијаграм секвенце размене порука у оквиру Веб апликације за визуализацију TLS протокола*



Слика 2*: Дијаграм случајева коришћења Веб апликације за визуализацију TLS протокола*

5.2. Имплементација решења

Имплементација криптографских механизама *TLS* протокола базира се на библиотекама програмског језика *python.* Библиотека која укључује криптографске алгоритме за извођење кључева, хешовање и шифровање, као и за манипулацију са сертификатима, је *cryptography.*

Шифровање блок шифрама *DES* и *AES*, реализовано је употребом *python* библиотеке *PyCrypto.* При томе су за генерисање кључа искоришћени *python* модули *binascii* и *os*, на следећи начин:

kljuc = binascii.hexlify(os.urandom(16))

Поред кључа, аргумент функције која имплементира *AES* алгоритам, је вектор иницијализације који гарантује другачији шифрат сваки пут када се порука шифрује. Вектор иницијализације одређен је на следећи начин:

vektor_inicijalizacije = ".join([chr(random.randint(0, 0xFF)) for i in range(16)]))])

Хеш-базирани код за аутентификовање порука (енг. *Hash-based Message Authentication Code*) добијен је помоћу *python* библиотека *hmac* и *hashlib*, где библиотека *hashlib* омогућава употребу различитих хеш функција: SHA-1, SHA-224, SHA-256, SHA-384, SHA-512.

Коришћење *RSA* алгоритма омогућују библиотеке PKCS1_OAEP и RSA, које припадају пакетима *Crypto.Cipher* и *Crypto.PublicKey*, респективно. *RSA* библиотека служи за генерисање сертификата и креирање пара јавни-приватни кључ. *PKCS1_OAEP*

библиотека служи за креирање кључа који се користи за шифровање поруке. У наставку су дате линије кода које представљају генерисање сертификата, јавног и приватног кључа, као и шифрата.

novi_kljuc = RSA.generate(4096, e=65537)

privatni_kljuc = novi_kljuc.exportKey("PEM")

javni_kljuc = novi_kljuc.publickey().exportKey("PEM")

kljuc = RSA.importKey(open('javni_kljuc.pem').read())

sifra = PKCS1_OAEP.new(kljuc)

sifrat = cipher.encrypt(poruka)

Python модул, *PyCryptodome*, садржи имплементацију криптографских алгоритама базираних на елиптичним кривама. У наставку је дат пример употребе ECC функције за генерисање кључа.

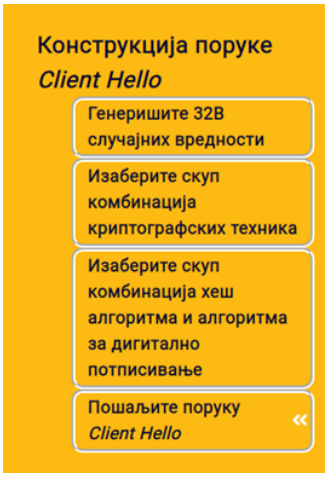
kljuc = ECC.generate(curve='P-256')

Различите елиптичне криве које је у оквиру ове функције могуће искористити су: secp192r1, sect233k1, secp224k1, secp256k1, NIST P-256, Curve25519, sect283k1, p384, secp384r1, sect409r1, Curve41417, Curve448-Goldilocks, M-511, P-521, sect571k1.

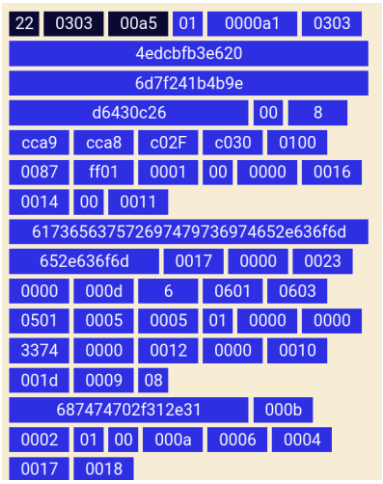
Визуелни део веб апликације омогућава интеракцију са корисником који диктира сваки корак протокола руковања између клијента и сервера. Након започињања визуализације протокола руковања, корисник је у могућности да диктира конструкцију сваке поруке која се у оквиру протокола размени. На Слици 3 приказане су опције које се кориснику нуде при конструкцији поруке *ClientHello*.

Свака од опција кориснику даје на увид стандардом прописане криптографске механизме којима клијент може да приступи серверу. Конструкцију сваке поруке одликује приказ бајтног записа поруке и објашњење значења сваког од поља у поруци. На Слици 4 дат је преглед свих поља поруке *ClientHello.*

Такође, након слања поруке која са собом носи материјал за генерисање кључа, при симболу клијента, односно, сервера, приказује се адекватан симбол и објашњење размењеног или генерисаног материјала кључа.



Слика 3: *Опције за конструкцију поруке ClientHello*



Слика 4: *Прозор који даје преглед свих поља поруке ClientHello*

6. ЗАКЉУЧАК

Познавање и разумевање криптографских протокола од изузетне је важности како за научнике који раде на побољшању ових протокола и инжењере софтвера, чија се професија умногоме заснива на безбедној електронској комуникацији, тако и за све кориснике рачунара и електронских услуга. У прилог томе, различите методе учења концепата криптографије и криптографских протокола обрађиване су у многим радовима [7] [8] [9], а у сврху успешнијег учења, развијени су многи алати. Циљ овог рада јесте да допринесе напорима да се олакша разумевање механизма функционисања TLS протокола. С тим у вези, рад најпре даје опсежан опис принципа на којима се заснива TLS протокол, као и опис дистинкције међу различитим верзијама протокола. Потом је у раду описана Веб апликација за визуализацију TLS протокола.

Кроз визуализацију TLS протокола у оквиру поменуте веб апликације, кориснику је омогућено вођење тока размене порука уз увид у бајтни запис сваке поруке. При томе се свака размена кључева или материјала за генерисање кључа, у оквиру апликације адекватно нагласи. Дакле, решење описано у овом раду, придружује се постојећим алатима исте сврхе, пружањем интерактивности са корисником као и наглашавањем кључних етапа размене порука у оквиру TLS протокола.

Описана постојећа решења укључују и демонстрацију напада на TLS криптографски протокол, чиме се додатно доприноси процесу његовог разумевања. Овај рад се ограничава на визуализацију самог механизма функционисања TLS протокола верзије 1.2, међутим, као део будућег рада, описана веб апликација покриће и демонстрацију напада на TLS протокол, али и остале његове верзије, пратећи принцип максималне интерактивности са корисником и нивоа детаљности огледане у бајтном запису.

7. ЛИТЕРАТУРА

[1] CrypTool Contributors (2021). JCrypTool. https://www.cryptool.org/en/jct/

[2] Driscoll, M., Denley, T., Mishra, M., Buhler, M. & Thomas, R. (2020, October 4). The Illustrated TLS Connection. https://tls.ulfheim.net/

[3] Oppliger, R. (2009). SSL and TLS: Theory and Practice (2nd ed.). Artech House.

[4] Thomas, S. (2000). SSL and TLS essentials. New Yourk.

[5] Ristić, I., (2017). Bulletproof SSL and TLS. London: Feisty Duck Limited

[6] Vachharajani, H. (2017). Implementation and Simulation of Secure Sockets Layer (SSL) in Windows Presentation Foundation (Doctoral dissertation).

[7] Yang, R., Wallace, L., & Burchett, I. (2011). Teaching cryptology at all levels using CrypTool. In Proc of the 15th Colloquium for Information Systems Security Education Fairborn (pp. 13-15).

[8] Hick, S., Esslinger, B., & Wacker, A. (2012). Reducing the complexity of understanding cryptology using CrypTool. In 10th International Conference on Education and Information Systems, Technologies and Applications (EISTA 2012), Orlando, Florida,USA.

[9] Adamović, S., Branović, I., Živković, D., Tomašević, V., & Milosavljević, M. (2011). Teaching interactive cryptography: the case for CrypTool. In IEEE Conference, ICEST.

Кратка биографија:

Милица Матијевић рођена је 1996. год. у Сомбору. Основне академске студије на Факултету техничких наука завршила је 2019. године. Мастер рад из области Електротехнике и рачунарства – Рачунарство и аутоматика, одбранила је 2021. године.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.42:004.738.12
DOI: https://doi.org/10.24867/15BE16Radovic

UPOTREBA WOLKABOUT PLATFORME ZA REALIZACIJU KUĆNOG IoT

USE OF WOLKABOUT PLATFORM FOR REALIZATION OF HOME IoT

Luka Radović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Zadatak rada predstavlja aplikaciju koja u saradnji sa Wolkabout platformom realizuje Home automation sistem. Serverski deo aplikacije je realizovan u programskom jeziku Java (Spring), a klijentski deo u najnovijoj verziji Angular okruženja. Aplikacija omogućava korišćenje Wolkabout platforme za CRUD operacije nad uređajima, senzorima i aktuatorima, ali predstavlja i brokera za slanje podataka (očitanih vrednosti senzora povezanog na Raspberry Pi) preko MQTT protokola na Wolkabout platformu koja je prikazuje stanje svega i menja stanja uređaja pod jasno definisanim pravilima. Na osnovu ranije očitanih temperaturnih vrednosti senzora, aplikacija obezbeđuje predikciju trenutne temperature.*

Ključne reči: *Internet of Things, Home automation, Wolkabout, Raspberry Pi*

Abstract – *This paper describes one implementation of Wolkabout platform for IoT Home automation system, with devices, sensors and actuators. Frontend is implemented in Angular and backend is implemented in Spring (Java). Application uses a real sensor for temperature and humidity connected to Raspberry Pi. Based on the previously read temperature values of the sensor, the application provides a prediction of the current temperature.*

Keywords: *Internet of things, Home automation, Wolkabout, Raspberry Pi*

1. UVOD

Zadatak rada predstavlja razvoj aplikacije koja u saradnji sa Internet of Things [1] Wolkabout platformom realizuje Home automation [2] sistem.

Internet of Things ili skraćeno IoT predstavlja tehnologiju za umrežavanje više uređaja različitog tipa, koji šalju podatke, na osnovu njih sprovode određene zadatke i to samostalno ili uz malu ljudsku intervenciju, kao i "razgovaraju" preko raznih komunikacionih protokola.

S obzirom da se radi o veb aplikaciji, ona se sastoji iz dva dela: klijentskog koji je implementiran u Angular [3] okruženju, kao i serverskog, za koji sam odabrao Java [4] programski jezik (Spring [5] okruženje). Aplikacija (LR Simulator) omogućava korišćenje Wolkabout platforme za CRUD [6] operacije nad uređajima, senzorima i aktuatorima, ali predstavlja i brokera za slanje podataka preko MQTT [7] protokola na Wolkabout platformu. Platforma služi za kreiranje uređaja, senzora, aktuatora, prikazivanje njihovih stanja, kao i za konfiguraciju pravila za automatizovan rad platforme nad uređajima, senzorima i aktuatorima. Podaci se čitaju iz Sqlite [13] baze podataka, a nastaju uz pomoć Python [14] programa koji očitava vrednosti DHT22 senzora (temperature i vlažnosti vazduha) koji je nakačen na Raspberry Pi 4 [15]. Takođe, moguće je i predvideti trenutnu temperaturu na osnovu ranije očitanih vrednosti senzora.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Milan Vidaković, red. prof.

2. OPIS KORIŠĆENIH TEHNOLOGIJA

2.1. Spring okruženje

Trenutno najpopularnije okruženje za razvoj Java veb aplikacija je Spring. Dependency injection [8] mehanizam za povezivanje objekata je jedna od njegovih glavnih karakteristika.

REST (Representational State Transfer) [9] je stil softverske arhitekture koji obezbeđuje standarde i olakšava komunikaciju sistemima na vebu. Glavne karakteristike su mu što je stateless (serveri koji ga implementiraju ne moraju ništa da znaju o klijentu, jer se on zasniva na postojanju resursa i uniformom upravljanju njima putem skupa predefinisanih operacija) i što je odvojena totalno implementacija na klijentu i serveru, tako da obe mogu biti menjane stalno, bez posledica, sve dok šalju poruke u dogovorenom formatu. Zahvaljujući REST – u, različiti klijenti mogu da "gađaju" iste servise, da koriste iste operacije i dobijaju iste odgovore.

Resursi su "imenice" veba - predstavljaju podatke ili funkcionalnosti identifikovane jedinstvenim identifikatorom (URI – Uniform Resource Identifier). Isti resurs može biti predstavljen u različitim formatima.

Komunikacija između klijenta i servisa u REST – u se zasniva na slanju request – a i response – a. Request treba da se sastoji od HTTP [10] metode (GET, POST, PUT, DELETE), zaglavlja koja daje više informacija o samom zahtevu, putanje do REST servisa i, po potrebi, telo poruke koja sadrži neke podatke. Zahvaljujući REST – u ti podaci mogu biti u raznim formatima, neki od njih su: image/png, audio/wav, application/json...

Spring Data JPA [11] (Java Persitence API) uvodi koncept repozitorijuma koji izbacuje potrebu za pisanjem ponavljajućeg koda. Repozitorijum je u stvari interfejs, kojim se upravlja podacima. Nije potrebno praviti klasu koja implementira interfejs, samim njegovim postojanjem je moguće koristiti operacije predviđene nasleđenim repozitorijumskim interfejsom (primer metode –

findById, koja pretražuje bazu po id – ju i vraća traženi objekat ili null vrednost ukoliko ne postoji). Postoje različiti tipovi repozitorijuma: CrudRepository, PagingAndSortingRepository, kao i JpaRepository koji sadrži standardne CRUD operacije, paginaciju i sortiranje, kao i mogućnost definisanja nestandardnih operacija nad podacima korišćenjem query anotacije napisanih u JPQL (Java Persistence Query Language). I dalje nema potrebe za implementacijom metode, samo je potrebno da se anotira željenim upitom. JpaRepository može biti bilo koja implementacija JPA. U mom projektu je to Hibernate [12], dok je za konkretan Data Source korišćena eksterna Sqlite baza podataka. Hibernate je okruženje koje pruža mogućnost mapiranja objektno orijentisanog modela na relacionu bazu podataka. Njegova glavna uloga je mapiranje javinih klasa na tabele u bazi.

2.2. Angular okruženje

Angular je okruženje za razvoj klijentskih aplikacija, koje su tipa Single Page Application, sastoje se iz komponenti koje se menjaju. Svaka komponenta je zadužena za deo funkcionalnosti. Rutiranje omogućava prelazak između njih.

Komponente objedinjuje element prikaza (HTML template), stilove koji se primenjuju na HTML i aplikativnu logiku (TS klasu) koja ga kontroliše. Nakon kreiranja komponente, ona mora biti uvrštena u deklaraciju NgModule – a. On predstavlja korenski modul aplikacije i obavezan je u svakoj.

S obzirom da svaka komponenta ima svoj prikaz, potrebno je imati navigaciju među njima, za tu svrhu postoji rutiranje, gde mi određujemo putanju i komponentu (prikaz) koji se otvara (primer: {path: "devices", component: DeviceComponent}).

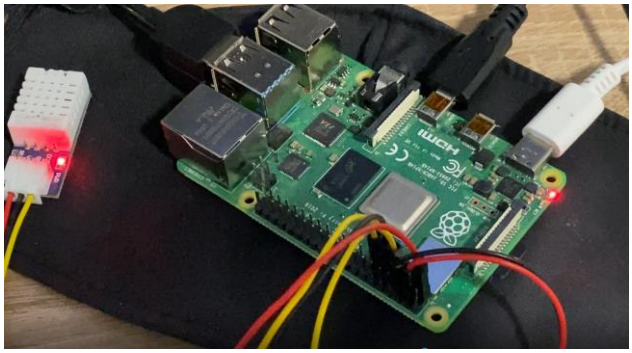
Još jedna od bitnih stvari Angular – a su servisi koji su spona između klijenta i servera. Njihova komunikacija se vrši preko Http protokola (Http Client biblioteke).

2.3. Raspberry Pi i DHT22

Raspberry Pi je mali kompjuter, veličine kreditne kartice koji može stati u džep. Cena mu je pristupačna, lako je prenosiv i može da se priključi na TV/monitor i da koristi standardnu tastaturu i miš. Nastao je u Velikoj Britaniji sa idejom da ljudima svih godina omogući da uče da programiraju na lak način. Zahvaljujući svojim mogućnostima, za kratko vreme postaje vrlo popularan te se ubacuje u različite grane privrede, jedna od njih je i IoT (Home automation).

Za ovaj projekat je korišćen Raspberry Pi 4 Model B sa 4 GB RAM - a. Kao što vidimo na slici 1, opremljen je sa: USB, micro HDMI, USB - C, ethernet, audio, 40 GPIO [16] pin - ova, WiFi - jem, Bluethooth - om, microSD karticom i tako dalje.

DHT22 je digitalni senzor temperature i vlažnosti vazduha (slika 1). Mana mu je što može da očitava vrednosti jednom u 2 sekunde, ne može u manjem intervalu. Ima 3 žice za povezivanje - power, digital out i ground, te se lako povezuje na Raspberry Pi. U mom slučaju, power žica je nakačena na 3.3v (1. pin), ground žica na ground (9. pin), a digital out na GPIO4 (7. pin). DHT22 očitava temperature od -40 do 80 stepeni Celzijusa, sa greškom od pola stepena, a od 0 do 100 procenata vlažnosti vazduha sa greškom od 2 do 5%.



Slika 1. *Raspberry Pi i DHT22*

2.4. Python

Python je jedan od najpopularnijih programskih jezika opšte namene. Jako je čitljiv zbog svoje sintakse, samim tim lako se uči i održavanje nije toliko skupo. Podržava korišćenje modula i paketa, te podstiče modularnost i ponovnu primenu istog koda. Pronalaženje i ispravljanje grešaka, popularnih "bugova", je jednostavno, ne postoji "segmentation fault", već kad interpreter pronađe grešku, baci izuzetak. Sam "debager" je napisan u Python - u.

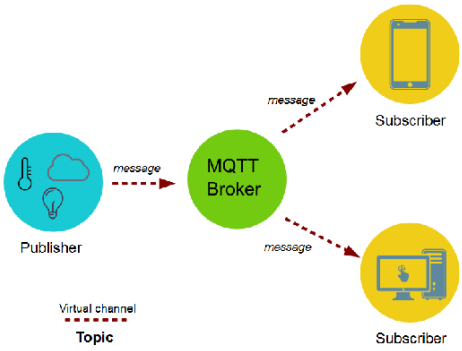
U projektu, Python je korišćen za rad sa senzorom, za dobijanje očitanih vrednosti i njihovih upisivanja u Sqlite bazu podataka. Za samo čitanje podataka mi je pomogla Adafruit_DHT [17] biblioteka i to svojom metodom read_retry, koja od parametara prima vrstu senzora i broj GPIO pin - a. Primer očitavanja vlažnosti vazduha i temperature upotrebom pomenute metode: h,t = dht.read_retry(dht.DHT22, DHT).

Python sam, takođe, koristio za predikciju trenutne temperature na osnovu ranije očitanih vrednosti senzora. Zahvaljujući klasi SensorData, imao sam sve što mi je potrebno za učenje, a to je timestamp i value - kada i šta je senzor očitao. Za rad sa bazom podataka, korišćena je biblioteka sqlite3. Predikciju sam realizovao pomoću polinomske regresije [18] i biblioteke sklearn, a da bih mogao da napravim REST servis, bila mi je potrebna flask biblioteka.

2.5. MQTT protokol

MQTT predstavlja publish/subscribe, extremno jednostavan i lak protokol za poruke, dizajniran za ograničene uređaje, visoka kašnjenja i ne toliko pouzdane mreže. Dizajnerski principi su da smanji količinu podataka koja se prenosi i resurse koje uređaji moraju da imaju, a takođe, pokušava i da osigura sigurnost dostave informacije.

Takvi principi su idealni za M2M ("machine-to-machine") ili "Internet of Things" svet pun uređaja, a i mobilnih aplikacija gde je količina podataka koja može da se prenese vrlo bitna, kao i trajanje baterije. Kao što vidimo na slici 2, MQTT protokol ima publishera koji šalje/objavljuje poruku na neku temu (Topic), a broker onda ima zadatak da je prosledi svim subscriberima te teme.

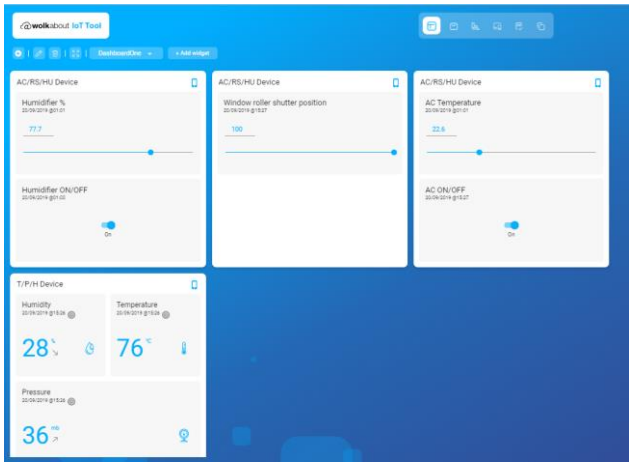


Slika 2. *MQTT protokol*

2.6. Wolkabout platforma

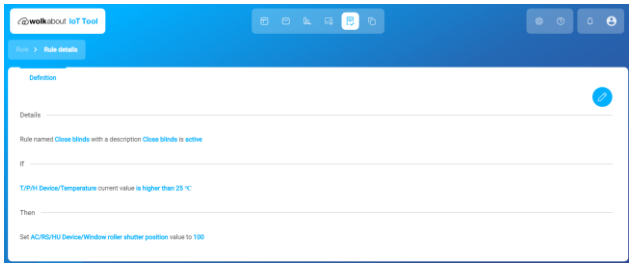
Na samoj Wolkabout platformi koja se nalazi na adresi (https://demo.wolkabout.com) korisniku je omogućeno da iskonfiguriše Dashboardove, kao i widgete i vidi stanje svojih uređaja, senzora i aktuatora.

Na slici 3 vidimo dva uređaja, jedan koji sadrži sve aktuatore i trenutne njihove vrednosti, kao i drugi koji sadrži sve senzore i vrši očitavanja, tačnije dobija podatke sa LR Simulatora.



Slika 3. *Wolkabout platforma*

Korisniku je omogućeno i da kreira pravila, po kojima određuje ponašanje uređaja u raznim situacijama. Na slici 4, vidimo pravilo Close binds. Ono podrazumeva spuštanje roletni ako temperatura bude iznad 25 stepeni. Moguće je i kombinovati više uslova.



Slika 4. *Definisanje pravila na platformi*

3. SPECIFIKACIJA APLIKACIJE

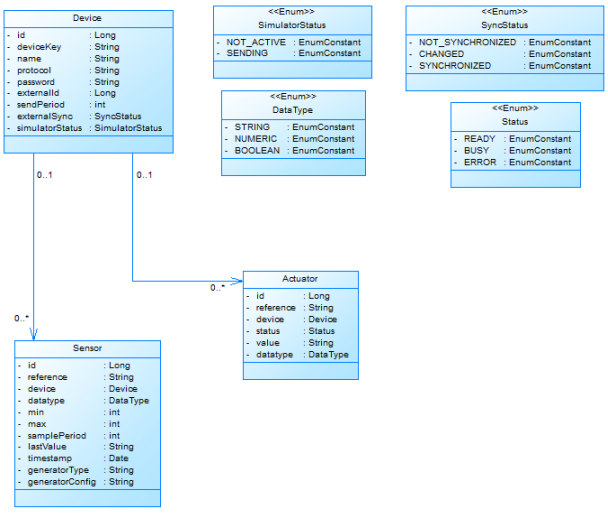
3.1. Dijagram klasa

Glavna klasa je Device, koja je sastavljena iz sledećih atributa: deviceKey (oznaka uređaja), name (ime), protocol (format poruke), externalId (id platforme), sendPeriod (na koliko vremenski šalje podatke), password (šifra), externalSync (da li je syncovana platforma sa našom bazom), kao i simulatorStatus (da li je pušteno slanje podataka).

Sledeća klasa je Sensor, neki od njenih atributa su: reference (jedinstveni identifikator senzora), device (kom uređaju pripada), dataType (enumeracija koja određuje tip podatka), min i max (vrednosti), lastValue (poslednja vrednost), timestamp (vreme očitavanja)...

SensorData klasa služi da opiše podatke koje očitava senzor, obuhvata: sensor (koji pokazuje koji senzor je očitao vrednost), device (kom uređaju taj senzor pripada), value (očitana vrednost) i timestamp (datum i vreme očitavanja).

Poslednja klasa modela je Actuator, koja je vrlo slična senzoru, a sastoji se iz: reference (jedinstveni identifikator), device (uređaj kome pripada), status (enumeracija u kom je stanju – READY, BUSY, ERROR), dataType (tip podataka sa kojima radi), value (vrednost sa kojom radi). Dijagram klasa prikazan je na slici 5.



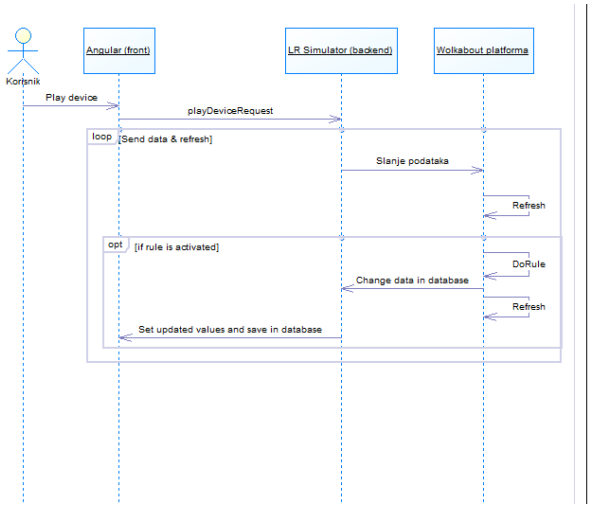
Slika 5. *Dijagram klasa*

3.2. Dijagram sekvence

Dijagram sekvence, kao što vidimo na slici 6, prikazuje pokretanje uređaja za slanje podataka. Ono, takođe, kreće od korisnika koji aktivira LR Simulator za slanje podataka određenog uređaja na platformu.

Wolkabout platforma prima podatke, obrađuje ih i osvežava vrednosti.

Ukoliko se prilikom nekog očitavanja ispuni uslov za pokretanje nekog pravila, platforma ga primenjuje i postavlja nove vrednosti.



Slika 6. *Dijagram sekvence*

4. ZAKLJUČAK

Internet of things je tehnologija koja je još uvek u razvitku i trebala bi tek da zablista u budućnosti. Home automation je sve češći i češći u domovima. Određeni proizvođači sve više ubacuju IoT funkcionalnosti u svoje uređaje, tako recimo LG frižideri javljaju svojim vlasnicima šta nedostaje i šta od namirnica treba da se kupi. To je samo jedna od sve učestalijih primena.

Budućnost IoT se ogleda u umrežavanju velikog broja uređaja u industriji i svakodnevnom životu, kao i njihovoj lakšoj kontroli i upravljanju sa udaljenih pozicija, kako bi se olakšao ubrzani život čovečanstva.

Što se tiče mog projekta, trebalo bi povezati još neki realan senzor/aktuator na Raspberry Pi i još više automatizovati svoj dom.

5. LITERATURA

[1] Internet of things
https://www.gkmit.co/blog/internet-of-things-iot-introduction-applications-and-future-scope
[2] Home automation
https://dzone.com/articles/home-automation-using-iot
[3] Angular okruženje
https://angular.io/
[4] Java
https://en.wikipedia.org/wiki/Java_(programming_language)
[5] Spring okruženje
https://spring.io/
[6] CRUD
https://www.codecademy.com/articles/what-is-crud
[7] MQTT protokol
http://mqtt.org/faq
[8] Dependecy injection
https://en.wikipedia.org/wiki/Dependency_injection
[9] REST
https://www.codecademy.com/articles/what-is-rest
[10]HTTP
https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol
[11] Spring Data
https://spring.io/projects/spring-data
[12] Hibernate
http://hibernate.org/
[13] Sqlite
https://www.sqlite.org/index.html
[14] Python
https://www.python.org/
[15] Raspberry Pi
https://www.raspberrypi.org/
[16] GPIO
https://en.wikipedia.org/wiki/General-purpose_input/output
[17] Adafruit
https://www.adafruit.com/
[18] Polinomska regresija
https://machinelearningmastery.com/polynomial-features-transforms-for-machine-learning/?fbclid=IwAR2pdyhj3CK5fkHzM9WuiKldOAMcIIBexXJXcU0E1Aa1tvCNMuYFSYqkbCA

Kratka biografija:

Luka Radović rođen je u Novom Sadu 1996. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva odbranio je 2019. godine. Od tada je zaposlen na FTN kao saradnik u nastavi.
kontakt: lukaradovic96@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/15BE17Radulovic

IMPLEMENTACIJA KONCEPTA DECENTRALIZOVANIH FINANSIJA ZASNOVANA NA PRIMENI POLKADOT ARHITEKTURE

IMPLEMENTATION OF DECENTRALIZED FINANCE CONCEPT BASED ON THE USE OF POLKADOT ARCHITECTURE

Danijel Radulović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu će biti predstavljeni koncepti blokčejna, decentralizovanih finansija i Polkadot blokčejn arhitekture. Biće opisana i konkretna implementacija koncepta decentralizovanih finansija zasnovana na primeni arhitekture Polkadot paralelnog lanca.*

Ključne reči: *distribuirani sistemi, blokčejn, decentralizovane finansije, Polkadot*

Abstract – *In this paper, the concepts of blockchain, decentralized finance and Polkadot blockchain architecture will be presented. The concrete implementation of the concept of Decentralized Finance based on the use of the Polkadot parallel chain architecture will also be described.*

Keywords: *distributed systems, blockchain, Decentralized Finance, Polkadot*

1. UVOD

U prošlosti je bilo mnogo pokušaja kreiranja digitalnog novca, ali su ti pokušaji uvek bili bezuspešni. Glavna prepreka kreiranja digitalnog novca je nedostatak poverenja. Ako bi digitalni novac postojao, kako osigurati da emitent digitalnog novca neće sebi dodeliti milion novčanih jedinica ili prisvojiti odnosno ukrasti digitalni novac za sebe. *Bitcoin*, jedna od najpoznatijih digitalnih kriptovaluta je kreirana na način da reši problem nepoverenja koristeći posebnu bazu podataka, koja je nazvana blokčejn.

Kako se blokčejn tehnologija razvijala, ideja digitalnog novca je konstantno evoluirala i donela novi koncept finansijskog sistema, tzv. decentralizovane finansije (engl. *DeFi*). Decentralizovane finansije se, kao globalni otvoren finansijski sistem, nameću kao alternativa tradicionalnom bankarskom sistemu. Izrada konkretne implementacija *DeFi* rešenja na *Polkadot* blokčejn arhitekturi će biti tema ovog rada.

2. TEORIJSKE OSNOVE BLOKČEJNA I DECENTRALIZOVANIH FINANSIJA

U ovom poglavlju, biće opisani koncepti koji čine teorijske osnove blokčejn tehnologije, počevši od osobina distribuiranih sistema do komponenata koje sačinjavaju jedan blokčejn sistem.

Takođe, biće obrađena i tema decentralizovanih finansija, koje se sve više ističu kao alternativa tradicionalnim bankarskim sistemima.

2.1. Blokčejn

Razumevanje distribuiranih sistema je od suštinskog značaja za razumevanje blokčejna, jer je blokčejn u svojoj osnovi distrubiran sistem. Tačnije, to je decentralizovani distribuirani sistem [1].

2.1.1. Distribuirani sistemi

Distribuirani sistem je skup autonomnih računara koji iz perspektive korisnika izgledaju kao jedan koherentan sistem. Ova definicija obuhvata dve glavne osobine distribuiranih sistema. Prva osobina ističe da je distribuirani sistem skup računara, od kojih svaki može da se ponaša nezavisno od ostalih. Druga osobina ističe da bi, iz perspektive korisnika, distribuiran sistem trebalo da deluje kao jedinstven sistem, tako da korisnici nisu svesni postojanja autonomnih računara [2].

S obzirom na prethodnu definiciju, distribuirani sistemi predstavljaju računarsku paradigmu, u kojoj dva ili više čvorova rade međusobno na koordiniran način radi postizanja zajedničkog ishoda, a da su modelovani na takav način da ih krajnji korisnici vide kao jedinstvenu logičku platformu [1].

Računarske jedinice, koje se češće nazivaju čvorovima (engl. *node*), mogu biti i hardverski uređaji i softverski procesi. Svi čvorovi su sposobni da šalju i primaju poruke između sebe. Čvorovi mogu biti ispravni, neispravni i maliciozni i mogu imati svoju memoriju i procesor. Glavni izazov u dizajnu distribuiranih sistema je koordinacija između čvorova i tolerancija na otkaze. Ako neki od čvorova postanu neispravni ili se mrežne veze prekinu, distribuirani sistem bi trebao tolerisati i nastaviti da radi bez smetnji, kako bi se postigao željeni rezultat.

Ovo je područje aktivnog istraživanja dugi niz godina i predloženo je nekoliko algoritama i mehanizama za prevazilaženje ovih problema.

Na osnovu dosadašnjeg opisa i objašnjenja vezanih za funkcionisanje distribuiranih sistema, mogu se izdvojiti njihove tri glavne karaksteristike:

- Konkurentnost komponenti – U distribuiranom sistemu je dozvoljeno da više klijenata istovremeno pristupi istom resursu, tako da i čvorovi mogu da izvršavaju više poslova istovremeno.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dušan Gajić, vanredni profesor.

- Nepostojanje globalnog sata – U distribuiranom sistemu ne postoji globalno vreme, tj. nisu svi satovi u sistemu sinhronizovani. Postoje algoritmi za sinhronizaciju, poput Berkli algoritma, koji skenira povremeno vrednost svih satova u sistemu, izračunava srednju vrednost i obaveštava sve članove o novom vremenu. S obzirom da često nije važno kada se događaj tačno desio, nego je bitan redosled događaja, uvodi se koncept logičkih satova, relacije desilo-se-pre i vremenskih otisaka.
- Nezavisan otkaz komponenti – Otkaz pojedinačnih komponenti neće uticati na rad sistema.

2.1.2. Komponente blokčejn sistema

Blokčejn je, u svom jezgru, *peer-to-peer* distribuirana glavna knjiga, koja je kriptografski sigurna, pri čemu se u nju može samo dodavati, ne može se izmeniti (izuzetno teško se menja) i ažurira se samo putem konsenzusa ili dogovora između *peer*-ova [1].

Važan pojam za objašnjenje ove definicije je pojam glavne knjige (engl. *ledger*). Glavna knjiga služi za beleženje i sumiranje ekonomskih transakcija. U glavnu knjigu se upisuju transakcije zajedno sa vremenskom oznakom, tako da u njoj postoji nedvosmislen skup svih transakcija u sistemu. Kada se na glavnu knjigu doda princip distribuiranosti, dolazi se do distribuirane glavne knjige, gde su podaci raspoređeni na više fizičkih mašina. Distribuirana glavna knjiga ili tehnologija distribuirane glavne knjige je vrsta distribuirane baze podataka koja pretpostavlja moguće prisustvo malicioznih korisnika (čvorova).

Za razliku od centralizovane glavne knjige koja obično ima centralizovano telo, koje vodi računa o validnosti glavne knjige, kod distribuirane glavne knjige svaki čvor može da upiše novu transakciju i zatim ostali čvorovi glasaju kroz konsenzus algoritam o tome koja kopija je validna. U trenutku kada je konsenzus postignut, svi čvorovi upisuju validnu verziju glavne knjige.

Blokčejn, kao struktura podataka, predstavlja lanac povezanih blokova. Blok predstavlja izbor transakcija zajedno povezanih radi njihove logičke organizacije. Sastoji se od transakcija i njegova veličina je promenljiva u zavisnosti od vrste i dizajna blokčejna koji se koristi. Svaki blok sadrži referencu na prethodni blok, osim ako nije u pitanju blok geneze. Blok geneze je prvi blok u blokčejnu, koji je postavljen kad je blokčejn pokrenut [1].

2.2. Decentralizovane finansije

Decentralizovane finansije (engl. skraćenica *DeFi*), u svojoj osnovnoj formi, predstavljaju sistem pomoću kojeg finansijski proizvodi postaju dostupni na javnoj decentralizovanoj blokčejn mreži, čineći ih otvorenim za svakoga, bez učešća posrednika, poput banaka i brokera. Za razliku od bankovnog ili brokerskog računa, lični dokument, broj socijalnog osiguranja ili dokaz adrese nisu potrebni za upotrebu *DeFi*-ja [3].

Preciznije, *DeFi* se odnosi na sistem pomoću kojeg softver napisan na blokčejnu omogućava kupcima, prodavcima, zajmodavcima i zajmoprimcima *peer-to-peer* interakciju sa striktno softverskim posrednikom, a ne sa kompanijom ili institucijom koja omogućava transakciju [3].

DeFi je otvoren i globalni finansijski sistem izgrađen za doba Interneta – alternativa sistemu koji je neproziran, strogo kontrolisan i koji zajedno drže više decenija stara infrastruktura i procesi. Omogućava korisnicima kontrolu i pregled nad svojim novcem, takođe omogućava izloženost globalnim tržištima i alternative lokalnoj valuti ili bankovnim opcijama. *DeFi* proizvodi otvaraju finansijske usluge svakome ko ima vezu ka Internetu i ceo sistem je u vlasništvu korisnika [4].

Većina *DeFi* aplikacija izgrađena na vrhu *Ethereum* mreže, druge po veličini svetske platforme za kriptovalute, koja se izdvaja od *Bitcoin*-a po tome što je lakše koristiti za izgradnju drugih vrsta decentralizovanih aplikacija, od jednostavnih transakcija.

Složenije slučajeve finansijske upotrebe je čak istakao i tvorac *Ethereum*-a, Vitalik Buterin, još 2013. godine u originalnom belom papiru *Ethereum*-a. Razlog tome su *Ethereum*-ovi pametni ugovori koji automatski izvršavaju transakcije ako su željeni uslovi ispunjeni, samim tim nude i mnogo više fleksibilnosti. Programski jezik *Solidity* je posebno dizajniran za kreiranje takvih pametnih ugovora.

U Tabeli 1 prikazano je poređenje između decentralizovanih i tradicionalnih finansija.

Tabela 1. *Poređenje DeFi-ja i tradicionalnih finansija*

Decentralizovane finansije	**Tradicionalne finansije**
Korisnik drži svoj novac	Korisnikov novac je u vlasništvu kompanije
Korisnik kontroliše kretanje i potrošnju svog novca	Potrebno je poverenje u kompanije da neće loše upravljati korisnikovim novcem, poput pozajmljivanja rizičnim zajmoprimcima
Transfer sredstava se vrši u nekoliko sekundi ili minuta	Plaćanja mogu trajati danima zbog manuelne obrade
Transakciona aktivnost je pseudoanonimna	Finansijska aktivnost je čvrsto povezana sa korisnikovim identitetom
DeFi je otvoren za svakoga	Korisnik se mora prijaviti za korišćenje finansijskih usluga
Tržište je otvoreno uvek	Tržišta se zatvaraju jer je zaposlenima potrebna pauza
Zasnovano na transparentnosti – svako može pogledati podatke o proizvodu i pregledati kako sistem funkcioniše	Finansijske institucije su „zatvorene knjige“ – ne možete tražiti da vidite njihovu istoriju kredita, zapis o upravljanju imovinom itd

U najpoznatije *DeFi* aplikacije spadaju [5]:

- Decentralizovane menjačnice (engl. skraćenica *DEX*) – Pomažu korisnicima da razmene valute za druge valute, bilo da su to američki dolari za *Bitcoin* ili *Bitcoin* za *Ethereum*. Trenutno su veoma popularna vrsta menjačnica, koja direktno povezuje korisnike kako bi mogli međusobno da trguju kriptovalutama bez posrednika, kojem bi poverili svoj novac.
- Stabilni novčići (engl. *Stablecoins*) – Kriptovalute koje su vezane za imovinu izvan kriptovaluta (npr. dolar ili evro), radi stabilizacije cene.
- Platforme za pozajmljivanje – Ove platforme koriste pametne ugovore umesto posrednika poput banaka koje upravljaju kreditiranjem.
- „Obmotan“ *Bitcoin* (engl. skraćenica *WBTC*) – Način slanja bitkoina na *Ethereum* mrežu, pružaju mogućnost korisnicima da zarađuju kamatu na *Bitcoin*-u, koji pozajmljuju putem gorenavedenih decentralizovanih platformi za pozamljivanje.
- Tržišta predikcije – Omogućavaju klađenje na ishod budućih događaja poput izbora.

Decentralizovane finansije su još u početnoj fazi svoje evolucije. Ukupna vrednost zaključana u *DeFi* pametnim ugovorima iznosi više od 84 milijarde dolara zaključno sa avgustom ove godine. Ukupna zaključana vrednost se računa množenjem broja tokena u protokolu sa njihovom vrednošću, izraženom u dolarima.

DeFi ekosistem je i dalje prepun infrastrukturnih grešaka, hakova i prevara. Jedna od najčešćih prevara predstavlja „povlačenje tepiha“ (engl. *rug pull*) , u kojima hakeri iscrpljuju sredstva iz protokola i onemogućavaju trgovanje investitorima. Takođe, postoje i dobro uspostavljeni protokoli, koji mogu značajno smanjiti rizik.

Otvorena i distribuirana priroda decentralizovanog finansijskog ekosistema može predstavljati i problem postojećim finansijskim propisima. Trenutni zakoni izgrađeni su na osnovu ideje o odvojenim finansijskim jurisdikcijama, od kojih svaka ima svoj skup zakona i pravila. *DeFi*-jev raspon transakcija bez granica predstavlja važna pitanja za ovu vrstu propisa, npr. ko je kriv za finansijski zločin, koji se događa preko granica, protokola i *DeFi* aplikacija [4].

Pametni ugovori su još jedno područje zabrinutosti za *DeFi* regulativu. Pored uspeha *Bitcoin*-a, *DeFi* je najjasniji primer teze „kod je zakon“, gde zakon predstavlja skup pravila, napisanih i primenjenih kroz nepromenljivi kod. S obzirom na to, greške su i dalje vrlo česte. Pametni ugovori su moćni, ali se ne mogu promeniti kada se pravila unesu u protokol, što često čini greške trajnim.

3. *POLKADOT* BLOKČEJN

Polkadot predstavlja mrežni protokol koji omogućava prenos proizvoljnih podataka, ne samo tokena, preko više blokčejn mreža. To znači da je *Polkadot* pravo okruženje za implementaciju interoperabilnosti između više različitih blokčejn mreža. *Polkadot* omogućava prenos podataka preko javnih, otvorenih i blokčejn mreža bez kontrole prava pristupa pa do privatnih mreža i blokčejn mreža sa kontrolom prava pristupa [6].

Tvorac *Polkadot*-a i suosnivač *Ethereum*-a, Gejvin Vud, je u belom papiru, opisao *Polkadot* kao skalabilni heterogeni višelančani blokčejn. Ovo znači da za razliku od prethodnih implementacija blokčejna, koji su se fokusirali na obezbeđivanje jedinstvenog lanca različitih stepena opštosti nad potencijalnim primenama, *Polkadot* je dizajniran tako da ne pruža nikakvu inherentnu funkcionalnost aplikacije, već pruža osnovni relejni lanac na kome veliki broj validnih i globalno koherentnih dinamičkih struktura podataka mogu biti korišćene rame uz rame.

Polkadot omogućava internet na kojem nezavisni blokčejnovi mogu razmenjivati informacije i transakcije preko relejnog lanca *Polkadot* mreže. Olakšava stvaranje i povezivanje decentralizovanih aplikacija, usluga i institucija.

Osnovne komponente od kojih je sačinjena *Polkadot* arhitektura su [7]:

- Paralelne niti (engl. *parathreads*) – Ideja za paralelne lance da privremeno (na nivou bloka) učestvuju u bezbednosti mreže, bez potrebe da iznajmljuju namensko mesto za paralelni lanac. Ovo se postiže ekonomičnim deljenjem resursa mesta za paralelni lanac među brojnim konkurentima, tj. paralenim nitima.
- Paralelni lanci (engl. *parachains*) – Struktura podataka specifična za aplikaciju, koja je globalno koherentna i koju mogu validirati validatori relejnog lanca. Najčešće paralelni lanci imaju oblik blokčejna, ali nema posebne potrebe za tim, tako da ne moraju biti u tom obliku. Zbog svoje paralelne prirode oni su u stanju da paralelizuju obradu transakcija i postignu skalabilnost *Polkadot* sistema. Takođe, učestvuju u bezbednosti cele mreže i mogu da komuniciraju sa drugim paralelnim lancima putem *XCMP*-a .
- Relejni lanac (engl. *relay chain*) – Centralni lanac *Polkadot*-a. Svi validatori *Polkadot* mreže zalažu svoja sredstva u relejni lanac u vidu *DOT* tokena. Relejni lanac sastoji se od relativno malog broja vrsta transakcija, kao što su interakcija sa mehanizmom upravljanja (engl. *governance*), aukcije za paralelne lance i učešće u postizanju konsenzusa. Glavna odgovornost je da koordinira sistem u celini, uključujući paralelne lance. Drugi specifični poslovi delegirani su paralelnim lancima, koji imaju različite implementacije i karakteristike
- Mostovi (engl. *bridges*) - Omogućavaju paralelnim lancima i paralelnim nitima da se povezuju i komuniciraju sa spoljnim mrežama poput *Ethereum*-a i *Bitcoin*-a.

4. KORIŠĆENE TEHNOLOGIJE

U implementaciji *DeFi* rešenja vezanog za ovaj rad, korišćeno je programsko okruženje *Substrate* i njegova integracija sa *Polkadot-JS* bibliotekom.

Ključni koncepti *Substrate*-a su *runtime*, *extrinsic*, apstrakcije naloga (engl. *Account Abstractions*), „bazen“

transakcija (engl. *Transaction Pool*), ključevi sesije (engl. *Session Keys*), težine transakcija (engl. *Transaction Weights*) i karakteristike van lanca (engl. *Off-chain features*). *Runtime* blokčejna je poslovna logika koja definiše njegovo ponašanje. U *Substrate* blokčejnovima, runtime se naziva "funkcija prelaska stanja", tu programeri definišu stavke skladišta koje se koriste za predstavljanje stanja blokčejna, kao i funkcije koje omogućavaju korisnicima blokčejna da unesu promene u to stanje [8]. Osnovna kodna baza *Substrate*-a isporučuje se sa *FRAME*-om, *Parity*-jevim sistemom za razvoj Substrate runtime-a, koji se koristi za lance poput *Kusama*-e i *Polkadot*-a. *FRAME* definiše dodatne primitive za *runtime* i pruža okruženje, koje olakšava konstrukciju *runtime*-a sastavljanjem modula, koji se nazivaju palete (engl. *pallets*). Svaka paleta obuhvata logiku specifičnu za domen, koja je izražena kao skup stavki skladišta, događaja, grešaka i otpremljivih funkcija, koje korisnici mogu pozivati [8].
Extrinsic-a predstavlja podatak koji dolazi izvan lanca, tj. mreže i uključen je u blok. *Extrinsic*-e se dele u tri kategorije: inherentne, potpisane i nepotpisane transakcije. *Extrinsic*-e su povezane zajedno u blok, kao niz koji se izvršava u redosledu definisanja extrinsic-e u toku vremena izvršavanja. Potpisane transakcije odgovaraju konceptu transakcije u *Ethereum*-u ili *Bitcoin*-u.
Polkadot-JS API je biblioteka interfejsa za komunkaciju sa *Polkadot* i *Substrate* čvorovima. *API* pruža mogućnost programerima aplikacija da postavljaju upite čvoru i stupaju u interakciju sa *Polkadot* i *Substrate* lancima, koristeći *Javascript*.

5. REŠENJE

Data implementacija *DeFi* rešenja predstavlja prototip *DeFi* platforme za pozajmljivanje sredstava, koja ima ulogu banke i praktično omogućava klasične tradicionalne usluge štednje i podizanja kredita. Implementacija ideje je zasnovana na konceptima *Polkadot* paralelnog lanca. Za izradu rešenja, kao osnova je iskorišćen već kreirani šablon *Substrate* čvora. *Substrate* projekat, kao što je ovaj, sastoji se od brojnih komponenti, koje su logički raspoređene u nekoliko direktorijuma. U *runtime* direktorijumu se nalazi datoteka za podešavanje i konfiguraciju *runtime*-a, definisanje vrednosti inicijalnih parametara paleta, kao i definisanje *RPC* metoda, koje korisnik može pozivati, kako bi dobio informacije o trenutnom stanju sistema. Konkretna *DeFi* paleta, koja predstavlja priloženo rešenje, sadrži pet *RPC* metoda: metodu za dobijanje trenutnog balansa korisnika sa obračunatom depozit kamatom, metodu koja vraća trenutni dug naloga sa obračunatom kamatom za pozajmicu, metodu koja vraća dozvoljenu visinu pozajmice za određenog korisnika, kao i metode za dobijanje vrednosti godišnje kamate na depozit i pozajmicu.
Od funkcionalnosti, tj. *extrinsic* metoda koje menjaju stanje sistema, podržan je depozit sredstava i stavljanje na štednju, povlačenje sredstava, uzimanje pozajmice i otplata duga.
U cilju poboljšanja ovog rešenja, jasno se nameće ideja uvođenja još kriptovaluta u kojima bi se mogle koristiti navedene usluge. Takođe, jedno od unapređenja bi moglo biti i uvođenje promenljivih kamata, kao i usluge brze pozajmice (engl. *flash loan*) koja predstavlja uzimanje i vraćanje pozajmice u okviru jednog bloka, što je nemoguće postići u tradicionalnom sistemu. Navedena poboljšanja predstavljaju, trenutno, sastavni deo svake veće *DeFi* platforme u blokčejn svetu.

6. ZAKLJUČAK

Blokčejn, kao tehnologija budućnosti, koja je pronašla svoje mesto u javnosti, zahvaljujući ideji digitalnog novca i kriptovaluta je započela revoluciju, od koje se očekuje da će se razvijati eksponencijalno. Širom sveta vlada veliko interesovanje za blokčejn tehnologijom od strane istraživača i raznih organizacija, što utiče veoma pozitivno na njen potencijal, konstantno donoseći nove koncepte i usavršavanje postojećih.
Decentralizovane finansije predstavljaju jednu od najbrže rastućih oblasti u blokčejn industriji. Centralizovani sistemi i ljudski faktori mogu ograničiti brzinu i sofisticiranost transakcija, dok korisnicima nude manje direktne kontrole nad novcem. Mnogi veruju da različiti *DeFi* projekti imaju veliki potencijal, privlačeći mnoštvo novih korisnika, čineći finansijske aplikacije inkluzivnije i otvorenije za one koji tradicionalno nemaju pristup takvim platformama.

LITERATURA

[1] I. Bashir, Mastering Blockchain, Packt, 2017.
[2] M. v. Steen i A. S. Tanenbaum, Distributed Systems, Prentice-Hall, 2017.
[3] R. Sharma, „Decentralized Finance (DeFi) Definition,“ 24.03.2021. [Na mreži]. Available: https://www.investopedia.com/decentralized-finance-defi-5113835 [Poslednji pristup 08.09.2021.]
[4] „Decentralized finance (DeFi),“ [Na mreži]. Available: https://ethereum.org/en/defi/ [Poslednji pristup 08.09.2021.]
[5] A. Hertig, „What Is DeFi?,“ 18.09.2020. [Na mreži]. Available:https://www.coindesk.com/tech/2020/09/18/what-is-defi/ [Poslednji pristup 08.09.2021.]
[6] „Polkadot - Technology,“ [Na mreži]. Available: https://polkadot.network/technology/ [Poslednji pristup 08.09.2021.]
[7] „Polkadot Wiki,“ [Na mreži]. Available: https://wiki.polkadot.network/ [Poslednji pristup 08.09.2021.]
[8] „Substrate Developer Hub - Knowledge Base,“ [Na mreži]. Available: https://substrate.dev/docs/en/ [Poslednji pristup 08.09.2021.]

Kratka biografija:

Danijel Radulović rođen je u Štutgartu, Republika Nemačka, 17. februara 1998. godine. Osnovne akademske studije je upisao na Fakultetu tehničkih nauka Univerziteta u Novom Sadu 2016. godine. Diplomirao je 2020. godine.
Kontakt: master.daca09@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.38:620.9
DOI: https://doi.org/10.24867/15BE18Milivojevic

PROGNOZA PROIZVODNJE FOTONAPONSKE ELEKTRANE NA OSNOVU METEOROLOŠKIH PARAMETARA

FORECAST OF PHOTOVOLTAIC POWER PLANT PRODUCTION BASED ON METEOROLOGICAL PARAMETERS

Marija Milivojević, Vladimir Katić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Predmet istraživanja ovog rada jeste praćenje prognoze meteoroloških parametara, stvaranje modela korelacije meteoroloških parametara i proizvodnje i određivanje modela za prognozu proizvodnje fotonaponske elektrane FTN1 u Novom Sadu.*

Ključne reči: *Solarna energija, Fotonaponska elektrana, Prognoza proizvodnje*

Abstract – *The aim of this paper is monitoring the meteorological parameters forecast, creating the correlation model of meteorological parameters and production and determination of the model of generation forecast.*

Keywords: *Solar energy, PV power plant, Generation forecast.*

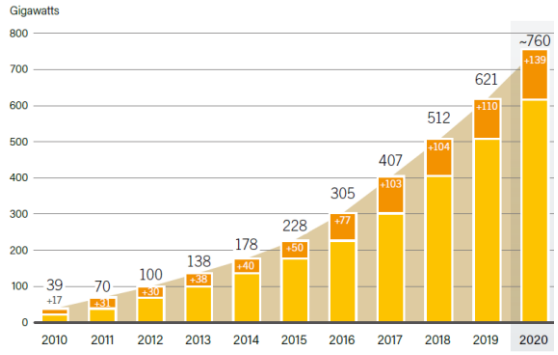
1. UVOD

Obnovljivi izvori energije (OIE) predstavljaju oslonac čovečanstva u borbi sa negativnim efektima gasova staklene bašte (posebno CO_2), koji se očituju u sve većim klimatskim promenama. Najčešće se koriste hidroenergija, energija vetra sunca, buomase, mora i geotermalna energija. Njihova transformacija u električnu energiju omogućava da se sve energetske potrebe ljudi obezbeđuju na ekološki prihvatljiv način i bez izraženih nuzefekata [1].

Za brzu popularizaciju i široko korišćenje, najpogodnija je solarna energija, koja se na jednostavan i jeftin način može transformisati u električnu ili toplotnu energiju. U ovom radu će biti razmatrana njena konverzija u električnu energiju, čiju osnovu čini foto-električni efekat, odnosno foto-naponska (FN) ćelija.

Rapidan napredak industrije i tehnologije proizvodnje FN ćelija poslednjih godina, rezultirao je tržišno prihvatljivim cenama komponenti FN sistema, a takođe i velikim rastom instalisanih kapaciteta.

Trenutno u svetu ima 760 GW FN elektrana sa dobrim daljim perspektivama razvoja (slika 1), dok se u Srbiji planira do 100 MW u narednih desetak godina [2,3]. Stoga je proizvodnja iz solarnih FN elektrana jedna od najperspektivnijih i najviše razvijajućih područja primene obnovljive energije.



Slika 1. *Instalisani kapaciteti FN elektrana u svetu [2]*

Sa tehno-ekonomskog aspekta, solarna energija je najbrže rastuća obnovljiva tehnologija i sektor sa najvećim stepenom investicija [4]. Ulaganje u solarne elektrane je isplativo, jer ova postrojenja imaju niske troškove održavanja, male zahteve za dodatno angažovanje radnika, visoku pouzdanost, dugotrajan životni vek (25-30 god.), te stabilan i predvidiv rad, pa sigurno donosi prihod [5].

Iz gore pomenutih razloga vidi se da FN elektrane dobijaju sve veći značaj u savremenim elektroenergetskim sistemima (EES) i da se njihovom pravilnom i pouzdanom radu poklanja sve veća pažnja. Jedan od aspekata rada u EES-u predstavlja i pouzdano planiranje proizvodnje bazirano na kratkoročnim ili dugoročnim prognozama.

U svetu je razvijeno više različitih metoda za prognoziranje rada FN elektrana, a tri najpoznatije su statistička-hronološka (statistical-time series), fizička (physical) i kombinovana (ensemble) metoda. One koriste savremene metode veštačke inteligencije (statistička), satelitskog snimanja i obrade slike (fizička) [6,7]. Međutim, one su veoma kompleksne, zahtevaju složene algoritme i precizne ulazne podatke. Stoga su potrebne neke jednostavnije metode prihvatljive tačnosti.

Cilj ovog rada je da se na osnovu posmatranja raspoloživih meteoroloških parametara utvrdi stepen njihove korelacije sa proizvodnjom FN elektrane, odrede najrelevantniji parametri i razvije dovoljno dobar model za prognozu proizvodnje FN elektrane koristeći kratkoročnu ili dugoročnu meteorološku prognozu.

2. SOLARNI POTENCIJAL SRBIJE

Broj časova sunčevog zračenja na teritoriji Srbije iznosi između 1.500 i 2.200 časova godišnje. Prosečan intenzitet sunčevog zračenja je od 1,1 kWh/m^2/dan na severu do 1,7 kWh/m^2/dan na jugu – tokom januara, a od 5,9 kWh/m^2/dan do 6,6 kWh/m^2/dan – tokom jula.

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

Prosečna vrednost energije zračenja iznosi od 1.200 kWh/m^2/godišnje u severozapadnoj Srbiji, do 1.550 kWh/m^2/godišnje u jugoistočnoj Srbiji, dok u centralnom delu iznosi oko 1.400 kWh/m^2/godišnje. Srbija ima znatno veći broj časova sunčevog zračenja nego većina evropskih zemalja, a najbolji uslovi su u jugoistočnom delu naše zemlje [8,9].

3. ZNAČAJ KVALITETNOG I JEDNOSTAVNOG MODELA PROGNOZE

Potrebe za preciznom prognozom proizvodnje FN elektrane povećavaju se razvojem FN elektrana i njihovim sve većim udelom u energetskim bilansima. Time se postiže povećanje pozdanosti i smanjenje troškova, sigurnije upravljanje elektroenergetskim mrežama, efikasnije trgovanje električnom energijom dobijenom iz FN elektrane i dr. [10]. Dodatno, predviđanje dovodi do smanjenja broja jedinica u stanju pripravnosti i smanjenju operativnih troškova rada u celom sistemu [11]. Posebno, ono predstavlja mogućnost adekvatnijeg upravljanja u slučaju agregiranja više manjih FN elektrana u virtuelne elektrane, jer se tada može umanjiti negativan uticaj promenljivosti sunčevog zračenja [11,12]. Takođe, na tržištu električne energije greška u prognozi može da ima velike finansijske posledice ukoliko znatno odstupa od prijavljenih količina proizvodnje za dan unapred.Zato je odvelike važnosti da prognostički modeli budu kvalitetni, odnosno da ih odlikuje tačnost i preciznost.

Osim što je kvalitet prognostičkog modela izuzetno bitan, od velikog je značaja i njegova jednostavnost. Cilj je da se stvori model koji zadovoljava po kvalitetu, a koji će funkcionisati sa malim brojem ulaznih parametara, odnosno meteoroloških podataka (temperatura, oblačnost, vlažnost, iradijacija, itd.). Veliki broj ulaznih parametara, zbog svoje brojnosti, predstavljaju opterećenje za računarski model, bazu podataka i zahtevaju veoma skup hardver.

Tačnost modela predviđanja proizvodnje FN energije može se povećati korišćenjem velikog broja ulaznih vektora. Međutim, računski troškovi i složenost će takođe biti povećani zbog agregacije velikog broja ulaznih parametara. Stoga je dizajniranje modela predviđanja sa optimalnim brojem ulaznih parametara na osnovu korelacije od najveće je važnosti [13].

4. PREDLOG MODELA ZA PROGNOZU PROIZVODNJE

Na osnovu podataka o solarnom potencijalu razmatrane lokacije, proračuna, ali i istorijskih podataka o promeni iradijacije tokom sunčanih sati u razmatranom periodu godine, kao i relevantnih meteoroloških podataka dobijenih prognozom, odnosno kombinacijom fizičkog i statističkog pristupa prognoze, može se uočiti određeni šablon promene solarne iradijacije i tako pretpostaviti vrednosti solarne iradijacije za neki određeni dan za koji treba izvršiti prognozu proizvodnje.

Te vrednosti, kao i vrednosti temperature vazduha, predstavljaju ulazne podatke u Matlab/Simulink model FN elektrane. Za ulazne podatke u model odabrani su upravo solarna iradijacija i temperature, jer se njihov uticaj na proizvodnju FN elektrane pokazao kao najznačajniji među svim ostalim meteorološkim parametrima, odnosno pokazali su najsnažniju korelaciju sa izlaznom snagom elektrane.

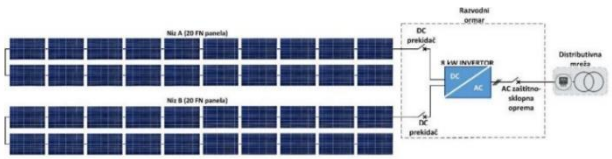
Dakle, model za prognozu proizvodnje FN elektrane podrazumeva određivanje, odnosno prognozu, ulaznih parametara u Matlab/Simulink model date elektrane (temperature vazduha i solarno zračenje). Zatim se, na osnovu unetih ulaznih podataka dobijaju izlazni podaci, odnosno tražena, prognozirana proizvodnja elektrane.

5. FN ELEKTRANA FTN1 KAO MESTO ZA PRIMENU I PROVERU METODA

Predloženi model za prognozu proizvodnje FN elektrane primenjen je, a zatim i proveren, u slučaju FN elektrane FTN1. Ova elektrana nalazi se u Novom sadu i puštena je u rad oktobra 2011. god., kao prva zvanično priključena FN elektrana na distributivnu mrežu EPS-a. Postavljena je na ravnom krovu zgrade Fakulteta tehničkih nauka (FTN), iznad dela u kojem se nalaze amfiteatri, biblioteka i čitaonica. Iako je raspoloživa površina krova 1.100 m^2, usled problema senčenja od strane okolnih objekata, samo je severni deo pogodan za postavljanje panela. Realizovana je fiksna noseća konstrukcija, tako da su paneli okrenuti prema jugu sa nagibom od 30° [14]. Lokacija FN elektrane FTN1 prikazana je na slici 2.

Slika 2. *Lokacija FN elektrane FTN1 i izgled ravnog krova na kom su postavljeni paneli [14]*

FN elektrana FTN1 direktno je povezana na distributivnu mrežu. Sastoji se od FN panela, invertora i zaštitne i sklopne opreme sa DC i AC strane. FN paneli organizovani su u dva niza od po 20 polikristalnih FN panela koji su postavljeni pod uglom od 30° u odnosu na površinu krova. Paneli su preko DC prekidača povezani na posebne ulaze invertora. Na AC strani invertora (ka mreži) nalaze se AC zaštitna i sklopna oprema i dvosmerno brojilo aktivne snage [14]. Struktura opisane FN elektrane data je na slici 3.



Slika 3. *Prikaz strukture FN elektrane FTN1 [14]*

Pojedinačne nominalne snage panela su 240 Wp, dok je ukupna instalisana snaga ove elektrane 9,6 kW, a nominalna izlazna snaga invertora je 8 kW [14].

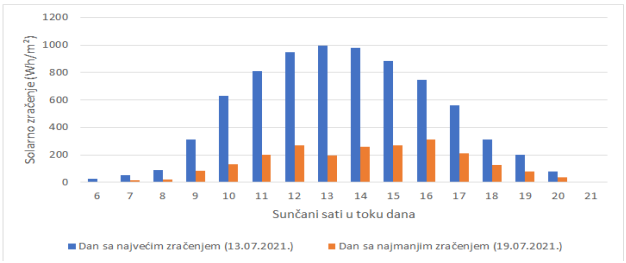
6. REZULTATI PRIMENJENOG METODA

Simulacija rada FN elektrane FTN1 izvršena je za period od sedam dana, počevši od 13.07.2021. i zaključno sa 19.07.2021. Tokom ovog perioda praćena je vremenska prognoza, odnosno meteorološki faktori, kao što su temperatura vazduha, brzina vetra, vlažnost, padavine i oblačnost, kao i solarna iradijacija na lokaciji razmatrane elektrane. Simulacija je izvršena pomoću Matlab/ Simulink modela FN elektrane FTN1 [15]. Kao ulazni podaci u model elektrane unose se izmerena solarna iradijacija i temperature vazduha, na osnovu kojih se kao odziv modela, odnosno izlazni podatak, dobija izlazna snaga FN elektrane.

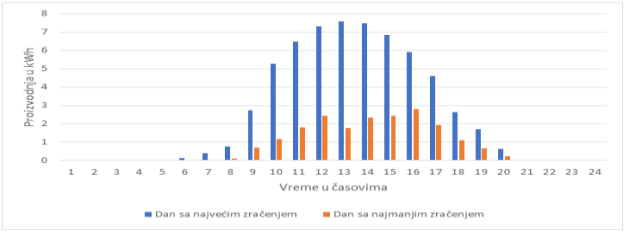
U razmatranom periodu, kao dan sa najvećim solarnim zračenjem, zabeležen je 13.07.2021. Ukupna vrednost intenziteta sunčevog zračenja u toku tog dana iznosila je 7,62 kWh/m². Najviša temperatura vazduha u periodu dnevne svetlosti, odnosno u periodu kada je bilo sunčevog zračenja na panele, iznosila je 38°C, a najniža 25°C.

Dan 19.07.2021. zabeležen je kao dan sa najmanjim sunčevim zračenjem ukupne vrednosti 2,206 kWh/m². Najviša temperatura vazduha u toku tog dana u period dnevne svetlosti iznosila je 25°C, a najniža 20°C.

Vrednosti solarnog zračenja tokom sunčanih sati i proizvodnja FN elektrane u danima sa najvećim i najmanjim zračenjem prikazane su na slikama 4 i 5, respektivno.



Slika 4. *Solarno zračenje tokom sunčanih sati u danu sa najvećim i najmanjim zračenjem*



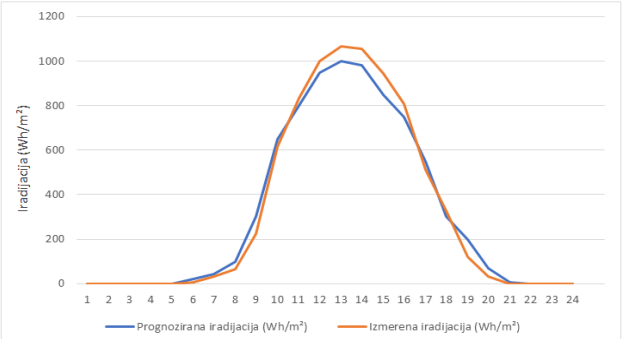
Slika 5. *Proizvodnja FN elektrane FTN1 dobijena simulacijom za dane sa najvećim i najmanjim solarnim zračenjem*

Na datim slikama se može uočiti da proizvodnja elektrane postoji isključivo tokom sunčanih sati i proporcionalna je iradijaciji.

7. PROCENA TAČNOSTI METODA

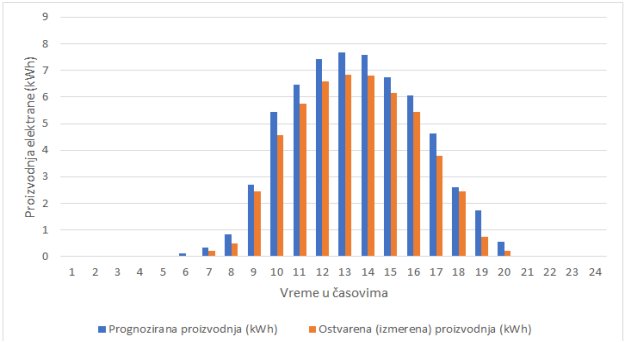
Sa ciljem provere verodostojnosti modela za prognozu proizvodnje FN elektrane FTN1 izvršeno je poređenje rezultata prognoze za određeni dan u budućnosti i ostvarene proizvodnje za isti dan. Podaci o ostvarenoj proizvodnji dobijaju se merenjem ostvarene proizvodnje na samoj FN elektrani koja se razmatra. Kao dan za koji je vršena prognoza i provera dobijenih rezultata izabran je 31.07.2021. Vremenska prognoza za dati dan izvršena je pet dana ranije pre izabranog datuma, u cilju prikupljanja meteoroloških podataka potrebnih za prognozu proizvodnje.

Na slici 6 vidi se odnos prognozirane i stvarne, tj. izmerene solarne iradijacije za dan 31.07.2021. Prognozirana vrednost iradijacije bila je nešto veća od izmerene u prvim i poslednjim sunčanim satima tog dana, a nešto manja od merene iradijacije u toku sati sa najvećom iradijacijom. Takođe, se može videti da su se u pojedinim časovima prognozirana i merena iradijacija gotovo podudarale.



Slika 6. *Prognozirana i izmerena iradijacija za dan 31.07.2021.*

Na slici 7 prikazane su prognozirana i ostvarena, tj. merena proizvodnja elektrane, dana 31.07.2021. Proizvodnja dobijena pomoću modela za prognozu proizvodnje tokom svih sunčanih sati razmatranog dana je nešto veća od ostvarene proizvodnje čije su merenjem na elektrani. U pojedinim časovima odstupanja su bila veća, a u pojedinim neznatna.



Slika 7. *Prognozirana i ostvarena (merena) proizvodnja za dan 31.07.2021.*

Razlozi zbog kojih se javljaju određena odstupanja merenih i prognoziranih vrednosti, jesu i odstupanja merenih i prognoziranih veličina koje se koriste kao ulazni parametri u model, tj. iradijacije, temperature ambijenta, ali i ostalih meteoroloških parametara koji utiču na proizvodnju FN elektrane.

Radi određivanja tačnosti predloženog prognostičkog modela proračunata je srednja kvadratna greška, na osnovu prognoziranih vrednosti proizvodnje koje su dobijene pomoću predloženog modela i podataka o ostvarenoj

proizvodnji dobijenih merenjem (dana 31.07.2021.). Srednja kvadratna greška (RMSE, *Root-Mean-Square Error*) određena je izrazom:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}\left(x_{i\_prognozirano} - x_{i\_mereno}\right)^2}{N}} \quad (1)$$

gde $x_{i\_prognozirano}$ označava prognoziranu proizvodnju elektrane za svaki sat tokom razmatranog dana, x_{i_mereno} označava merenu vrednost ostvarene proizvodnje elektrane za svaki sat tokom tog dana, a sa N je označen ukupan broj sati tokom dana kada je vršeno merenje.

Srednja kvadratna greška izračunata prema izrazu (1) iznosi 0,504.

Naravno, ova greška, a time i tačnost modela, dosta zavise od tačnosti meteorološke prognoze, koja može biti različita (pa i nedopustivo velika), a na osnovu koje se biraju vrednosti ulaznih parametara modela. Primenom predloženog modela prognoze za dan 31.07.2021., vidi se da je predviđena proizvodnja veća u odnosu na ostvarenu (slika 7). To odstupanje zavisi i od tačnosti meteorološke prognoze, koja u pojedinim danima može biti veća ili manja, pa to predstavlja rizik predložene metode. Kao posledica predviđanja veće proizvodnje od ostvarene, može doći do debalansa u EES, tj. dodatnih troškova u sistemu, pa i zahteva za plaćanje penala kao nadoknade štete kod kupaca električne energije. Za detaljniju analizu i ocenu nivoa pomenutog rizika, potrebno je posmatrati duži vremenski period (i do godinu dana), što je van opsega ovog rada, odnosno plan za dalje istraživanje.

8. ZAKLJUČAK

U ovom radu pokazano je da je moguće koristiti modele za prognozu proizvodnje FN elektrane sa ograničenim (manjim) brojem ulaznih parametara, koji su javno dostupni. Na osnovu rezultata simulacije, kao i na osnovu poređenja prognoziranih vrednosti proizvodnje, koji su dobijeni kao rezultat predloženog modela za proizvodnju, i merenih vrednosti ostvarene proizvodnje FN elektrane FTN1, predloženi prognostički model se ocenjuje kao dovoljno dobar za predviđanje približne proizvodnje FN elektrane za različite vremenske prilike.

Zaključuje se da je model FN elektrane FTN1 verifikovan, da dovoljno dobro opisuje stanje i ponašanje razmatrane elektrane i da se kao takav može koristiti u prediktivne svrhe, ali samo za približno predviđanje. Ipak, na osnovu proračunate greške, zaključuje se da je, kada je u pitanju tržište električne energije, veća tačnost prognoze ipak od velikog značaja.

9. LITERATURA

[1] V. Katić, I. Kapetanović, N. Sarajlić, "Obnovljivi izvori električne energije", TEMPUS-CEFES, Fakultet tehničkih nauka, Novi Sad, 2007.

[2] ***, „Renewables 2021 – Global status report", REN21, Paris, 2021.https://www.ren21.net/wp-content/uploads/2019/05/GSR2021_Full_Report.pdf

[3] **, „Strategija razvoja energetike Republike Srbije do 2025. godine sa projekcijama do 2030. godine", Sl. Glasnik RS br.101/2015, Beograd, 2015.

[4] ***, „Global landscape of Renewable Energy Finance 2020", IRENA, Abu Dhabi, 2020.

[5] G. Reikard, „Predicting solar radiation at high resolutions: a comparison of time series forecasts", Solar Energy, 2009; 83: pp.342–9.

[6] S. Sobri et al., „Solar photovoltaic generation forecasting methods: A review", Energy Conversion and Management, Vol.156, 2018, pp.459–497.

[7] Ž. Stojanović, V.A. Katić, "Kratkoročna prognoza proizvodnje fotonaponske elektrane", Zbornik radova FTN, God. 36, br. 3, 2021, pp.468-471.

[8] T. Pavlović et al., „Possibility of electricity generation using PV solar plants in Serbia", Renewable and Sustainable Energy Rev., Vol.20, 2013, pp.201-218.

[9] http://www.hidmet.gov.rs/latin/meteorologija/klimatologija_srbije.php

[10] Connecting the Sun-Solar Photovoltaics on the Road to Large Scale Grid Integration. EPIA, 2012.

[11] J. Antonanzas, et al., "Review of photovoltaic power forecasting", Solar Energy, Vol.136, 2016, pp.78–111.

[12] Mills, A.; Wiser, R. Implications of Wide-Area Geographic Diversity for Short-Term Variability of Solar Power; Technical Report LBNL-3884E; Lawrence Berkeley National Laboratory: Washington, DC, USA, Sept. 2010.

[13] U.K. Das, et al., „Forecasting of photovoltaic power generation and model optimization: A review", Renewable and Sustainable Energy Review, Vol.88, 2017, pp.912-928

[14] V.A. Katić, et al., „Realizacija krovne fotonaponske elektrane na Fakultetu tehničkih nauka u Novom Sadu", Tehnika, God.64, br.4, 2015, pp.655-662

[15] Marija Joković, Vladimir Katić "Tehnike rekonstrukcije u fotonaponskim elektranama u slučaju senčenja panela", Zbornik radova FTN, God. 36, br. 3, 2021, pp.464-467.

Kratka biografija:

Marija Milivojević, dipl.inž. rođena je 1995. god. u Novom Sadu. Srednju školu Gimnazija, završila je u Inđiji, 2013. god. Fakultet tehničkih nauka, studijski program Energetika, elektronika i telekomunikacije (OAS) upisala je školske 2013/2014. Na studijama se opredelila za modul Elektroenergetika 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) i završila ih septembra 2021. god.

Vladimir A. Katić, red.prof. rođen je 1954. god. u Novom Sadu. Diplomirao je na Fakultetu tehničkih nauka u Novom Sadu 1978. god., a magistrirao i doktorirao na Univerzitetu u Beogradu 1981. i 1991. god., respektivno. Od 2002. god. je redovni profesor Univerziteta u Novom Sadu, a oblasti interesovanja su energetska elektronika, obnovljivi izvori električne energije, kvalitet električne energije i električna vozila.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/15BE19Ilic

ANALIZA TAČNOSTI IMPEDANTNE METODE PRI VARIJACIJI PARAMETARA DISTRIBUTIVNE MREŽE

ACCURACY ANALYSIS OF THE IMPEDANCE METHOD IN VARIATION OF PARAMETERS OF DISTRIBUTION NETWORK

Danojla Ilić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu razmatran je proračun kratkih spojeva i lokacije kvara primjenom impedantne metode u distributivnim mrežama, priložene teorijske osnove o istim, a takođe definisan i algoritam pomoću koga se dolazi do rješenja zadatog problema. Analiziran je uticaj dužine sekcije na tačnost proračuna, zatim preciznost proračuna kod kvarova na početnim sekcijama izvoda, kao i uticaj uzemljenja transformatora. Obrađen je i uticaj dodatih generatora na pojedinim izvodima na rezultate proračuna. Svi proračuni su realizovani pomoću programa koji je za potrebe ovog rada razvijen u programskom jeziku C++. Dobijeni rezultati su prikazani tabelarno i izvršena je njihova analiza.*

Ključne reči: *Distributivna mreža, Impedantna metoda, Lokacija kvara*

Abstract – *This document presents the calculation of short circuits and fault locations using the impedance-base method in distribution networks, the attached theoretical basis for the same, and also defines the algorithm by which to solve the given problem. The influence of the section length on the accuracy of the calculation, then the accuracy of the calculation in case of failures on the initial sections of the terminal, as well as the influence of the grounding of the transformer were analyzed. The influence of added generators on individual leads on the calculation results is also treated. All calculations are done by using a program specially designed for the purposes of this thesis, developed in the C++ programming language. The obtained results are presented in the form of tables and they were appropriately analyzed.*

Keywords: *Distribution network, Impedance-based method, Fault location*

1. UVOD

Elektroenergetski sistem je tehnički sistem čiji je osnovni zadatak da obezbjedi pouzdano i kvalitetno napajanje električnom energijom.

Elektroenergetski sistemi su konstantno izloženi kvarovima, što utiče na njihovu pouzdanost, sigurnost i kvalitet električne energije.

U ovom radu, analizirana je tačnost impedantnog algoritma za procenu lokacije kvarova, uslijed promjene parametara distributivne mreže [1-5].

U drugoj glavi ukratko su opisani elementi elektroenergetskog sistema, sa posebnim osvrtom na generatore, transformatore i vodove.

U trećoj glavi su priložene teorijske osnove o kratkim spojevima i detaljno jeopisan admitantno-impedantni algoritam.

U četvrtoj glavi opisani su algoritmi za metodologije koje se koriste za procjenu mjesta kvara, njihove prednosti i mane.

Peta glava opisuje primjenu impedantnog algoritma na izabranoj test mreži. U cilju detaljnije analize vršena je promjena parametara distributivne mreže i analiziran uticaj istih na tačnost proračuna.

Poslednje dve glave su zaključak i literatura.

2. ELEMENTI ELEKTROENERGETSKIH SISTEMA

Elementi elektroenergetskih sistema su generatori, transformatori, vodovi i potrošači. U ovoj glavi opisani su generatori, transformatori i vodovi.

2.1. Elektrane

Podsistem proizvodnje sastoji se od elektrana koje predstavljaju osnovne izvore električne energije, koji treba da u svakom trenutku zadovolje potrošnju sistema, kao i da obezbjede neophodnu rezervu kapaciteta za pokrivanje iznenadnih promjena opterećenja i neplaniranih ispada proizvodnih kapaciteta.

2.1.1. Generatori

Pored elektrana podsistem proizvodnje čine i generatori koji se priključuju na distributivne mreže (35 kV, 20 kV i 10 kV) – distributivni generatori (DG). Takođe, generatori priključeni na niskonaponsku mrežu dio su podsistema proizvodnje – DG u okviru samih potrošača. U poslednje vrijeme oni postaju sve atraktivniji i pomoću njih je moguće rasteretiti kapacitete osnovne proizvodnje elektrana.

Problemi koji se javljaju priključenjem DG na mrežu su ostrvski rad, skraćenje dosega releja i problem selektivnosti zaštite.

2.2. Transformatori

Transformatori električnu energiju jednog napona i struje pretvaraju u električnu energiju nekog drugog napona i struje. Njihov rad se zasniva na elektromagnetnoj indukciji.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Duško Bekut, red. prof.

Na osnovu eksperimenta kratkog spoja i praznog hoda, određuju se parmetri pogonskih šema trofaznih dvonamotajnih transformatora.

Ako su namotaji trasformatora tako povezani i priključci izvedeni i njegovo priključenje u elektroenergetski sistem u simetričnom režimu ne poremeti simetriju režima, tada se kaže da taj transformator ima određenu spregu.

Sprega se označava kombinacijom slova i brojeva, gdje je prvom slovo veliko i predstavlja način povezivanja visokonaponskog (VN) namotaja, a drugo slovo je malo i predstavlja način povezivanja niskonaponskog (NN) namotaja.

Sprežni broj čine brojevi od 0 do 11, gdje se povezivanjem namotaja VN i NN na isti način dobijaju sprege sa parnim, a povezivanjem na različit sprege sa neparnim sprežnim brojevima.

2.3. Trofazni vodovi

Sekcije vodova mogu biti nadzemne i podzemne sa izolovanim i neizolovanim provodnicima. U distributivnoj mreži podzemni vodovi su sa izolovanim provodnicima, a nadzemni su sa neizolovanim provodnicima. Upotreba jednih ili drugih zavisi od više faktora kao što su potrebna pouzdanost napajanja, razvijenost sredine i drugi.

3. KRATKI SPOJEVI

Pod kratkim spojem se podrazumijeva direktan spoj između tačaka koje se u normalnim uslovima nalaze na različitim potencijalima, pri čemu vrijednosti struja kratkog spoja znatno premašuju nominalne.

3.1. Impedantno – admitantni algoritam

Distributivna mreža se dekomponuje na kolo sa režimom prije kvara i fiktivno kolo.

Proračun u fiktivnom kolu počinje učitavanjem podataka o distributivnoj mreži (DM), numeracijom grana i čvorova po lejerima, a zatim i računanjem dva ekvivalenta:

- ekvivalenta mreže do krajeva i
- ekvivalenta mreže do korijena mreže.

Sledeći korak je proračun režima sa ratkim spojem na mjestu kratkog spoja, gdje je potrebno odrediti tip i mjesto istog.

Za zadati tip kratkog spoja vrijednosti simetričnih komponenti struja na mjestu kratkog spoja su funkcija napona na mjestu kratkog spoja prije kratkog spoja – koji se dobija se pri proračunu tokova snaga i odgovarajuće Thevenin-ove impedanse koja se računa na osnovu ekvivalentne impedanse.

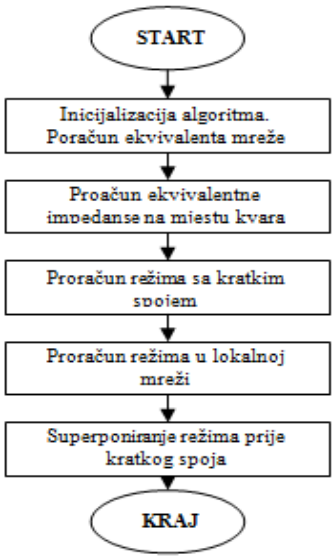
Na slici 1 prikazan je globalni blok dijagram admitantno – impedantnog algoritma za model mreže.

4. PROCJENA MJESTA KVARA

Da bi se kvar uspješno eliminisao, neophodno ga je prvo locirati, a zatim i izolovati od ostalog dijela DM, a nakon toga i izvršiti restauraciju napajanja.

4.1. Metodologije za procjenu mjesta kvara

Lokacija kvara zahtijeva određeno vrijeme u toku koga potrošači ostaju bez napona.



Slika 1. – *Globalni blok dijagram admitantno – impedantnog algoritma*

U otkrivanju dionice sa kvarom značajnu ulogu imaju detektori kvara na osnovu kojih se saznaje kroz koje dionice je protekla struja kvara. Postupak se izvodi kroz više iteracija od početka izvoda ka kraju. Ako je detektor aktivan, to znači da je kroz to mesto „prošla“ struja kvara. Kada se pronađe stanica u kojoj je detektor deaktiviran, proces se zaustavlja. Kvar se nalazi na dionici između stanice sa detektorom koji je odreagovao i deaktiviranim detektorom.

Neke od metoda lokacije kvara bazirane su na praktičnim iskustvima, kao što su utvrđivanje okoline kvara na osnovu većeg broja poziva potrošača koji se nalaze u relativnoj blizini.

Postupak metode polovljenja nije potreban ako je distributivna mreža opremljena SCADA nadgledanjem i ako su svi potrebni uređaji povezani sa SCADA upravljanjem.

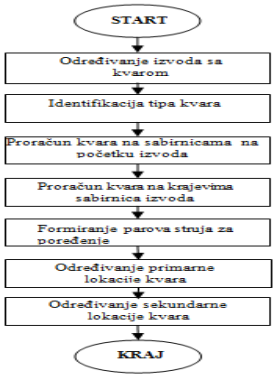
Najkvalitetnije procjene se dobijaju kombinovanjem upotrebe računarske metode i brze mjerne jedinice. To je slučaj kod strujnog i impedantnog algoritma.

4.1.1. Strujni algoritam

Strujni algoritam zasnovan je na mjerenju struje. Postupak počinje određivanjem izvoda i tipa kratkog spoja, a zatim se ustanovljeni tip kvara (u ovom radu jednopolni kvar= simulira na sabirnicama sa kojih polazi izvod sa kvarom, a onda i na sabirnicama na krajevima svih dionica tog izvoda.

Za svako mjesto kvara računa se vrijednost struje koja bi se dobila mjerenjem brzom mjernom jedinicom, formirajući na taj način uređen par vrijednosti struja koje odgovaraju vrijednostima struja za kratki spoj na početku i na kraju date dionice.

Nakon toga se upoređuju izračunate vrijednosti struje sa mjerenim vrijednostima, za sve dionice. Za moguće dionice sa kvarom se uzimaju one kod kojih je mjerena vrijednost struje manja ili jednaka od prve, a istovremeno veća ili jednaka od druge vrijednosti uređenog para vrijednosti struja. Na slici 2. prikazan je globalni blok dijagram strujnog algoritma.



Slika 2. – *Globalni blok dijagram strujnog algoritma*

4.1.2. Impedantni algoritam

Impedantna metoda lokacije kvara počinje određivanjem ekvivalentne impedanse direktnog i nultog redoslijeda na osnovu režima prije kvara, a zatim se nakon detektovanja kvara računa ekvivalentna impedansa petlje kvara koja zavisi od tipa kvara. Na osnovu date impedanse se locira kvar.

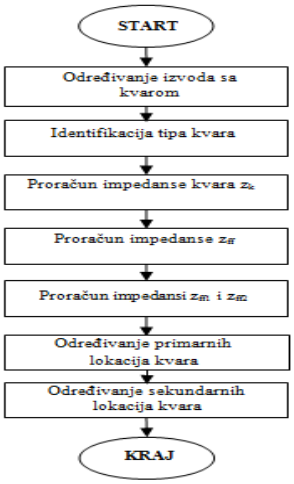
Na osnovu izmjerenih vrijednosti napona i struje, računa se impedansa između mjerne jedinice i mjesta kvara. Koji tip kvara je u pitanju, saznaje se na osnovu mjerenja, identifikacijom faza sa povećanim vrijednostima struje. Na kom izvodu se dogodio kvar se saznaje na osnovu prorade releja. Distantni relej mjeri impedansu direktnog redoslijeda.

Primarne dionice sa kvarom su one kod kojih je imaginarni dio impedanse Z_{ff} istovremeno manji ili jednak od imaginarnog dijela impedanse Z_{ff2}, a veći od imaginarnog dijela impedanse Z_{ff1}, tj za koje važi sledeća relacija:

$$Im\{Z_{ff1}\} \leq Im\{Z_{ff}\} \leq Im\{Z_{ff2}\}..... \quad . \qquad (1)$$

gdje su Z_{ff1} i Z_{ff2} impedanse dionice na početku i kraju, a Z_{ff} izračunata impedansa.

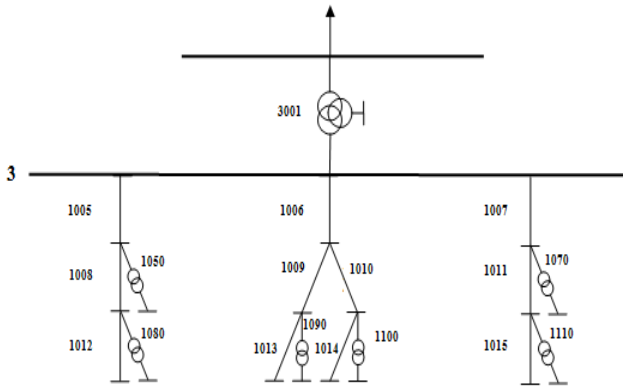
Uzimajući u obzir grešku mjerenja procjenjuju se i moguće dionice koje predstavljaju sekundarne lokacije kvara. Na slici 3. prikazan je globalni blok dijagram impedantnog algoritma.



Slika 3. – *Globalni blok dijagram impedantnog algoritma*

5. PRIMJENA IMPEDANTNOG ALGORITMA

U ovoj glavi je dat primjer proračuna lokacije određenog tipa kratkog spoja primjenom impedantnog algoritma na test mreži prikazanoj na slici 4.



Slika 4. – *Jednostavna distributivna mreža*

Svi proračuni su napravljeni sa podrškom razvijenom u programskom jeziku C++.

Na ovoj mreži potrebno je pronaći lokaciju kvara impedantnom metodom. Zatim treba simulacijom kvara na određenim sekcijama i promjenom određenih parametara, analizirati preciznost metode: kod veoma kratkih sekcija, početnih sekcija, kao i uticaj uzemljenja transformatora na tačnost proračuna. Na kraju je potrebno dodati generator na drugi kraj mreže i analizirati njegov uticaj na tačnost određivanja mjesta kvara.

Ulazni podaci o jednostavnoj DM smješteni su u odgovarajućim ulaznim fajlovima, a vrednosti su preuzete iz [1].

Prije početka proračuna potrebno je normalizovati parametre DM. Nakon toga se numerišu grane i čvorovi po lejerima i DM se priprema za tretman kratkih spojeva.

5.1. Uticaj dužine sekcija na tačnost proračuna

U ovom primjeru se sekciji 1008 mjenja inicijalna dužina u svrhu analize tačnosti, pa se zadaju redom dužine 100 m, 50 m, 20 m i 10 m, a zatim se analizira preciznost proračuna ukoliko se kvar desi na 20 %, 40 %, 60 % i na 80 % dužine sekcije. Tačnost detektovanja deonice izvoda sa kvarom pri apsolutno tačnim mjerenjima prikazana je u tabeli 1, gdje oznaka T^{1008} pokazuje da je tačno detektovana dionica sa kvarom, a superskript 1008 predstavlja oznaku dionice, dok analogno tome, npr N^{1012} pokazuje da nije detektovana tačna dionica sa kvarom, već dionica 1012.

Tabela 1. – *Tačnost detektovanja dionice sa kvarom pri različitim dužinama dionice i mjesta kvara za apsolutnu tačnost mjerenja*

	1 [km]	0.1 [km]	0.05 [km]	0.02 [km]	0.01 [km]
0.2	T^{1008}	T^{1008}	T^{1008}	T^{1008}	T^{1008}
0.4	T^{1008}	T^{1008}	T^{1008}	T^{1008}	T^{1008}
0.6	T^{1008}	T^{1008}	T^{1008}	T^{1008}	T^{1008}
0.8	T^{1008}	T^{1008}	T^{1008}	T^{1008}	T^{1008}

Međutim, ako se uzme u obzir greška mjerenja od +1% dobijaju se rezultati prikazani u tabeli 2.

Što je dionica kraća, a mjesto kvara udaljenije od početka dionice, to je veća nepreciznost detektovanja lokacije kvara.

Tabela 2. – *Tačnost detektovanja dionice sa kvarom pri različitim dužininama dionice i mjesta kvara uzimajući u obzir grešku mjerenja od +1%*

	1 [km]	0.1 [km]	0.05 [km]	0.02 [km]	0.01 [km]
0.2	T^{1008}	T^{1008}	T^{1008}	T^{1008}	N^{1012}
0.4	T^{1008}	T^{1008}	T^{1008}	T^{1008}	N^{1012}
0.6	T^{1008}	T^{1008}	T^{1008}	N^{1012}	N^{1012}
0.8	T^{1008}	T^{1008}	N^{1012}	N^{1012}	N^{1012}

5.2. Preciznost proračuna kvarova na početnim sekcijama

U ovom dijelu simulira se kvar na sekcijama 1005, 1006 i 1007. Pogrešna dionica sa kvarom, u ovom slučaju, detektuje se samo kada se mjesto kvara nalazi pri kraju dionice i uz velike greške mjerenja, što je i prikazano u tabeli 3. Simulirani su kvarovi na sekciji 1005 (čija je dužina 1 km) i to na 90%, 93%, 96% i 99% dužine sekcije, uzimajući u obzir da je greška mjerenja +2%, +4%, +6%, +8% i +10%.

Tabela 3 – *Tačnost proračuna na početnim sekcijama izvoda uzimajući u obzir grešku mjerenja i udaljenost mjesta kvara*

	1.02	1.04	1.06	1.08	1.10
0.90	T^{1005}	T^{1005}	T^{1005}	T^{1005}	T^{1005}
0.93	T^{1005}	T^{1005}	T^{1005}	N^{1008}	N^{1008}
0.96	T^{1005}	T^{1005}	N^{1008}	N^{1008}	N^{1008}
0.99	N^{1008}	N^{1008}	N^{1008}	N^{1008}	N^{1008}

5.3. Uticaj uzemljenja transformatora na tačnost proračuna

Vrijednost rezistanse otpornika R_{uz} preko koga je transformator uzemljen je sada je 40Ω umjesto 0Ω. Pokretanjem proračuna detektuje se tačna dionica kvara. Povećanjem R_{uz} na 200Ω, takođe se dobijaju tačni rezultati, što znači da povećanje rezistanse ne utiče na tačnost proračuna.

5.4. Uticaj račvanja dionica na tačnost proračuna

Kvar se simulira na sekciji 1009. Ukoliko su svi parametri inicijalni, a mjesto kvara na 80% dužine dionice, pokretanjem proračuna dobijaju se dva moguća rješenja, sekcije 1009 i 1014.

To znači da u slučaju račvanja impedantni algoritam nije dovoljno precizan, pa je potrebna primjena i detektora kvara koji se stavlja na račve i određuje na kojoj strani se nalazi kvar.

5.5. Dodavanje generatora na krajevima test mreže i njegov uticaj na proračun

U test mrežu se dodaju generatori u čvorove 20, 6 i 14. Analiziraju se različiti slučajevi, prva tri kada je aktivan po jedan generator i četvrti kada su aktivna sva tri generatora. Analizira se kvar na sekciji 1008 na 20%, 40%, 60% i 80% dužine sekcije. Rezultati su u tabeli 4. iz koje se vidi da i kada je aktivan jedan i kada su aktivna sva tri generatora detektuje se tačna dionica.

Tabela 4. *Tačnost detektovanja dionice sa kvarom pri različitim dužininama dionice u slučaju dodavanja generatora u izabranim čvorovima*

	Aktivan generator u čvoru 20	Aktivan generator u čvoru 6	Aktivan generator u čvoru 14	Aktivna sva tri generatora
0.2	T^{1008}	T^{1008}	T^{1008}	T^{1008}
0.4	T^{1008}	T^{1008}	T^{1008}	T^{1008}
0.6	T^{1008}	T^{1008}	T^{1008}	T^{1008}
0.8	T^{1008}	T^{1008}	T^{1008}	T^{1008}

6. ZAKLJUČAK

Cilj ovog rada bio je da se uradi analiza tačnosti primjene impedantne metode u lokaciji kvarova, kao i koliko na to utiču pojedini parametri. Da bi se to moglo realizovati, bilo je neophodno na izabranom primeru mreže, korišćenjem impedantno-admitantnog algoritma analizirati tačnost impedantne metode za lociranje kvara na različitim primerima.

Do grešaka dolazi najviše uslijed grešaka mjerenja, na početnim sekcijama izvoda i to ako je mjesto kvara blizu kraja sekcije. Takođe, kod veoma kratkih sekcija, što je kvar udaljeniji, uslijed pozitivnih grešaka mjerenja, dolazi do pogrešnog detektovanja dionica, dok za negativne greške mjerenja važi suprotno. Pokazano je da povećanje vrijednosti rezistanse otpornika, preko koga je transformator uzemljen, ne utiče na preciznost proračuna.

Ustanovljeno je da račvanje rezultuje sa više mogućih rješenja, gdje su jednake šanse za izbor ciljne dionice, pa se taj problem može riješiti upotrebom detektora kvara. Na kraju je utvrđeno da i dodavanje generatora na krajevima mreže ne utiče na proračun, što je bio očekivan rezultat.

7. LITERATURA

[1] Dragan Popović, Duško Bekut, Valentina Dabić: „Specijalizovani DMS algoritmi“, DMS Group, Novi Sad, 2011.

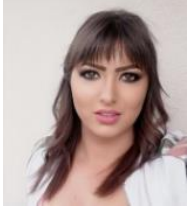
[2] Vladimir Strezoski: „Analiza elektroenergetskih sistema“, Fakultet tehničkih nauka, Novi Sad, 2012.

[3] M. D. Nimrihter, P. N. Đapić: „Proračuni u distributivnim električnim sistemima“, Fakultet Tehničkih nauka, Novi Sad, 2008.

[4] Milan S. Ćalović, Predrag Č. Stefanov, Andrija T. Sarić: „Eksploatacija elektroenergetskih sistema u uslovima slobodnog tržišta“, Tehnički fakultet, Čačak.

[5] https://www.researchgate.net /publication/337857354 UTICAJ_DISTRIBUIRANIH_GENERATORA_NA_DI-STRIBUTIVNU_MREZU

Kratka biografija:



Danojla Ilić rođena je u Bileći 1993. god. Osnovne studije završila je na Fakultetu tehničkih nauka 2017. god iz oblasti Elektrotehnike i računarstva, smjer Elektroenergetski sistemi. Master studije upisala je 2017. god na istom fakultetu smjer Elektroenergetika – Elektroenergetski sistemi.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/15BE20Stepanovic

ZAŠTITA ELEKTRANE NA BIOGAS

BIOGAS POWER PLANT PROTECTION

Stefan Stepanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu predstavljen je primer zaštite elektrane na biogas. Navedeni su osnovni principi proizvodnje elektrane na biogas. Objašnjena je mikroprocesorska zaštita koja se koristi u elektrani na biogas. Opisane su generalno zaštite transformatora i generatora u elektranama na biogas. Dat je primer elektrane na biogas i prikazane su specifičnosti i osobine zaštite transformatora i generatora.*

Ključne reči: *Elektrana na biogas, Zaštita generatora, zaštita transformatora*

Abstract – *This document presents an example of biogas power plant protection. The basic principles of biogas power plant production are stated. The microprocessor protection used in a biogas power plant is explained. The protections of transformers and generators in biogas power plants are generally described. An example of a biogas power plant is given and the specifics and characteristics of transformer and generator protection are presented.*

Keywords: *Biogas power plant, Generator protection, Transformer protection*

1. UVOD

Danas je poznato više obnovljivih izvora energije. Energija koja se dobija iz bioloških izvora se naziva energija biomase.

Produkt koji se dobija iz digestije biomase jeste biogas. Biogas se dobija se bakterijskom razgradnjom organskog materijala u anaerobnom digestoru u elektrani na biogas. Obično se koristi za dobijanje dva glavna proizvoda: biogas za proizvodnju električne energije i digestat koji se koristi kao dodatak za poboljšanje kvaliteta zemljišta ili visokokvalitetno đubrivo [1, 2].

Da bi elektroenergetski sistem mogao da funkcioniše potrebno je obezbediti zaštitu svakog njegovog elementa (generator, transformator, vod, itd). Upravo zaštita celokupnog elektroenergetskog sistema od kvarova se ostvaruje relejnom zaštitom [3].

Cilj ovog rada jeste da prikaže zaštitu, njene specifičnosti i osobine u jednoj elektrani na biogas, prvenstveno zaštitu najbitnijih jedinica kao što su generator i transformator. U drugoj glavi opisan je biogas, njegove osobine i karakteristike, kao i ciklus proizvodnje biogasa.

U trećoj glavi date su osnovne teze o mikroprocesorskoj zaštiti koja je osnova zaštite u biogasnim elektranama. U nastavku glave je opisana zaštita generatora uopšteno, šta je povod za njeno korišćenje i koje vrste zaštite postoje i koje su njihove osobine. U četvrtoj glavi opisana je zaštita transformatora generalno i data je podela na vrste koje postoje. U petoj glavi dat je primer elektrane na biogas koja je obrađena i zaštita koja se koristi u toj elektrani na biogas i njene specifičnosti. U šestoj glavi dat je zaključak rada, a u sedmoj glavi korišćena literatura.

2. BIOGAS

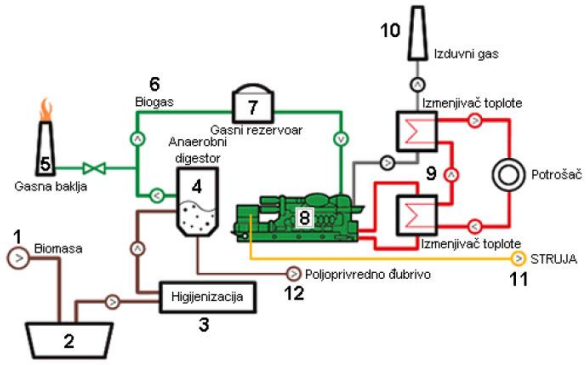
Biogas se sastoji iz mešavine metana (50 – 75 %), ugljen dioksida (25- 50 %) i ostalih gasova kao što su azot, kiseonik, vodonik sulfid (2 – 8 %). Procenat metana u biogasu je glavna komponenta za njegovo korišćenje kao izvor energije. Metan sagoreva mnogo čistije u odnosu na ugalj. [1].

2.1. Proces proizvodnje elektrane na biogas

Proces proizvodnje biogasa, kao i faze u proizvodnji električne energije sastoje se od nekoliko koraka: sakupljanje supstrata i njegovo skladištenje, predtretman supstrata, anaerobna digestija, predtretman biogasa i proizvodnja električne energije [1].

U prvoj fazi u procesu se vrši sakupljanje i skladištenje supstrata u rezervoaru za privremeno skladištenje. Tipovi supstrata koji se najviše koriste su stajnjak, kukuruzna silaža, agroindustrijski otpad, komunalni otpad. U drugoj fazi dolazi do čišćenja na 70°C gde se uništavaju sve bakterije negativne po proces digestije. Anaerobna digestija je glavna faza u procesu proizvodnje biogasa. Ona se realizuje u zavisnosni od vrste bakterija i temperaturnih uslova. Uobičajeno, digestori rade u mezofilnim uslovima i potrebno je da temperatura bude konstantna i pH vrednost od 6,5 do 7,5. Potrebno je da se u potpunosti ostvare anaerobni uslovi, jer najmanja količina kiseonika dovodi do umiranja bakterija i prekida procesa digestije. U četvrtoj fazi biogas mora biti prečišćen postupkom desumporizacije i sušenja. Desumporizacija se primenjuje za sprečavanje toksičnog dejstva vodonik sulfida. Sušenjem se sprečava potencijalna kondenzacija vodene pare kojom je biogas zasićen nakon izlaska iz digestora. U petoj fazi biogas se iz gasnog rezervoara prosleđuje na SUS motor, gde se stvara toplota tokom rada motora koja se efektivno koristi preko izmenjivača toplote. Korišćenjem generatora, mehanička energija gasnog motora se pretvara u električnu energiju. Na slici 1 je dat vizuelni prikaz procesa proizvodnje biogasa i električne energije elektrane na biogas [1, 2, 4].

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je dr Duško Bekut, redovni profesor



Slika 1. *Proces proizvodnje biogasa i električne energije elektrane na biogas [2]*

3. O ZAŠTITI GENERATORA GENERALNO

Za zaštitu u elektrani na biogas se koriste mikroprocesorski releji koja predstavlja jednu savremenu grupu releja. Mikroprocesorske zaštite mogu biti povezane sa SCADA sistemom, tako da se pri radu mogu pratiti i promene u uklopnom stanju u postrojenju i spram tih promena prilagođavati akcije. Nakon promene parametara, mikroprocesorskoj zaštiti je potrebno obično između jedne i dve sekunde da bi se nastavilo funkcionisanje sa novim parametrima. Savremene mikroprocesorske zaštite obično imaju banke pohranjenih podataka za neke ili svaki od parametara. Korišćenje podataka iz tih banaka je izuzetno jednostavno – dovoljno je na određeni ulaz dovesti signal, pa će tekući parametri biti zamenjeni odgovarajućim iz banaka.

Kvarovi na generatoru nastaju kao posledica proboja izolacije izazvanog različitim naprezanjima. To su mehanička naprezanja zbog centrifugalnih sila, dinamičkih sila kratkih spojeva, vibracija kao i naprezanja pri grejanju i hlađenju. Izolacija je izložena i termičkim i hemijskim uticajima zbog kojih dolazi do starenja i gubitka izolacionih osobina. Osim toga izolacija može biti izložena i previsokim električnim naprezanjima. Do njih može doći zbog atmosferskih ili pogonskih prenapona. Teži kvarovi generatora imaju za posledicu velike materijalne izdatke za popravku ili zamenu oštećenog dela, dok su štete izazvane nemogućnošću proizvodnje električne energije još veće [3].

Prema vrsti i mestu, razlikuju se sledeći kvarovi generatora:

- kratak spoj između namota statora,
- kratak spoj sa zemljom statorskog namota,
- kratak spoj između navojaka jedne faze statorskog namota,
- kratak spoj sa zemljom pobudnog namota i
- kratak spoj između navoja pobudnog namota [3].

Kao zaštite od kvarova kod generatora u elektranama na biogas koriste se sledeće zaštite:

- zemljospojna zaštita statora,
- zemljospojna zaštita rotora,
- prenaponska zaštita statora,
- zaštita od povratne sprege,
- termička zaštita namotaja,
- podfrekventna zaštita,
- nadfrekventna zaštita,
- prekostrujna zaštita,
- naponski kontrolisana prekostrujna zaštita,
- diferencijalna zaštita generatora.

3.1. Zemljospojna zaštita statora

Zemljospojna zaštita statora zavisi od načina uzemljenja zvezdišta generatora, odnosno veličine generatora. Za generatore malih snaga (<2 MW) obično su direktno uzemljeni, ili su uzemljeni preko niskoomske impedanse ili otpornika. U tim slučajevima, zemljospojna zaštita statora izvodi se kao zemljospojna vremenski nezavisna zaštita. Ova zaštitna funkcija koristi se za detekciju struja zemljospoja: strujni transformator postavljen u zvezdištu generatora, odnosno povratni vod tri strujna transformatora na faznim priključcima generatora. Vremensko kašnjenje omogućava selektivnost zaštite i njeno usaglašavanje sa drugim zaštitama u sistemu [4].

3.2. Prenaponska zaštita statora

Prenaponi na priključcima generatora najčešće se javljaju zbog kvara na regulatoru pobude, kvara na regulatoru regulacionog blok–transformatora ili naglog rasterećenja. Prenapon može poticati od generatora ili od sistema, što se određuje praćenjem smera reaktivne snage. Ako prenapon potiče od sistema, treba isključiti samo visokonaponski prekidač, a ako potiče od generatora, treba izvršiti potpuno isključenje. Zaštita može da bude jednofazna ili trofazna. Prenaponska zaštita se koristi sa vremenskim kašnjenjem, čime se sprečava nepotrebno reagovanjem pri kratkotrajnim skokovima napona koji se javljaju pri naglim promenama opterećenja [4].

3.3. Zaštita od povratne sprege

Generator ulazi u motorni režim rada kada se dotok pare ili vode ka turbini prekine, a jedinica ostane vezana na mrežu. To može da bude posledica normalnog isključenja turbine, ili reagovanja turbinske zaštite, ili reagovanja kotlovske zaštite. Zaštita se zasniva na praćenju smera aktivne snage i obično se realizuje u dva stepena: prvi stepen isključuje generator sa kratkim vremenskim kašnjenjem ako je turbinski ventil zatvoren, a drugi stepen isključuje generator sa dužim vremenskim kašnjenjem i ako turbinski ventil nije zatvoren. Zaštita može biti realizovana sa jednofaznim, dvofaznim ili trofaznim merenjem, i zahteva strujne i naponske ulaze. Za zaštitu od povratne sprege se primenjuju odgovarajući releji snage kojima se deluje na isključenje generatora [4].

3.4. Termička zaštita namotaja

Ova zaštitna funkcija štiti namotaje generatora od povećanja temperature usled povećanja struja opterećenja. Algoritam ove funkcije računa procentualno povećanje temperature namotaja na osnovu izmerene struje i poznatih podataka: nazivne struje i vremenske konstante zagrevanja mašine koja se štiti. Prilikom proračuna mogu da se uzimaju u obzir i uslovi hlađenja. Ova zaštita ima dva izlaza, jedan za alarm, a drugi za isključenje. Može da bude jednofazna ili trofazna. Zaštitom bi trebalo da se isključi generator nešto pre nego što namot dostigne maksimalno dozvoljenu temperaturu [4].

3.5. Naponski kontrolisana prekostrujna zaštita

Ova funkcija služi za zaštitu generatora od spoljašnjih kratkih spojeva. Sadrži logičku kombinaciju dve funkcije: prekostrujne i podnaponske. Prekostrujna funkcija se

podešava na vrednost koja je veća od maksimalne radne struje, a manja od minimalne struje kratkog spoja. Pri spoljašnjim kratkim spojevima struja kvara brzo opada što može dovesti do nepoželjnog resetovanja zaštite. Da bi se to sprečilo, ova zaštita memoriše maksimalnu vrednost struje kvara nakon reagovanja i pamti je određeno vreme u toku koga treba da reaguje i podnaponska funkcija. Ako podnaponska funkcija ne reaguje u zadatom vremenu, prekostrujna funkcija se resetuje.

Podnaponska funkcija se podešava na vrednost koja je manja od nominalnog napona, a veća od napona pri pojavi kratkog spoja. (tipično 0,75 Un). Zaštita ima dodatno vremensko kašnjenje, čime se postiže usaglašavanje sa drugim prekostrujnim zaštitama. Zaštita se resetuje posle davanja signala za isključenje ili nakon ponovnog uspostavljanja napona sistema [4].

4. ZAŠTITA ENERGETSKIH TRANSFORMATORA

Kvarovi na transformatorima nastaju kao posledica slabljenja i oštećenja izolacije. Ta oštećenja mogu biti izazvana naprezanjima električne prirode (nastaju kao posledica atmosferskih i pogonskih prenapona u mreži), mehaničke prirode (izazvane dinamičkim silama u namotima) ili kao posledica prevelikih zagrevanja. Pomenuta naprezanja dovode do polakog i neminovnog starenja i slabljenja izolacije i do pojave kvarova. Kada su u pitanju kvarovi kod transformatora postoji zahtev za brzom eliminacijom kvara jer postoji opasnost od pucanja transformatorskog suda – kotla i paljenja ulja. Kod transformatora se sreću sledeći kvarovi:

- Kratki spojevi između namota transformatora.
- Kratki spojevi između navojaka iste faze.
- Kratki spojevi sa zemljom (bilo namota, bilo izvoda namota)
- Lokalna tinjanja izolacije zbog previsokih električnih naprezanja ili kao posledica smanjenja kvaliteta izolacije usled prevelikog zagrevanja.

Za zaštitu energetskih transformatora u elektranama na biogas koriste se sledeće zaštite:

- buhloc zaštita,
- prekostrujna zaštita,
- kratkospojna zaštita,
- zemljospojna zaštita,
- termička zaštita.

5. ZAŠTITA ELEKTRANE NA BIOGAS

U elektrani na biogas se koriste mikroprocesorski uređaji, kao samostalni releji ili u okviru sistema integrisane zaštite i upravljanja elektranom. Sva zaštitna oprema mora da radi nezavisno od rada sistema upravljanja, nadzora i komunikacije u okviru elektrane. U slučaju nestanka pomoćnog napona za napajanje zaštitnih uređaja i strujnih krugova komandi rasklopnih aparata u elektrani, dolazi do automatskog isključenja elektrane sa distributivne mreže na spojnom prekidaču.

5.1. Specifičnosti zaštite elektrane na biogas

Za zaštitu generatora i elemenata rasklopne aparature elektrane od mogućih havarija i oštećenja usled kvarova i poremećaja primenjuju se sledeće zaštite : sistemska zaštita i zaštita priključnog voda. Delovanjem ovih zaštita mora se na spojnom prekidaču izvršiti automatsko prekidanje paralelnog rada elektrane sa distributivnog sistema električne energije. Pored ovih zaštita se koristi dodatna zaštita koja se priključuje u poseban razvodni orman i koju čini vazdušni prekidač snage čija je naznačena struja 2500 A, dok je naznačena simetrična struja prekidanja 65 kA. On obuhvata zaštitu od preopterećenja, trenutnu prekostrujnu zaštitu, zemljospojnu zaštitu, termičku zaštitu. Izgled prekidača je prikazan na slici 2.

Slika 2. *Spoljašni izgled vazdušnog prekidača snage*

Sistemska zaštita se sastoji od naponske zaštite i frekventne zaštite. Naponska zaštita reaguje na poremećaj ravnoteže između proizvodnje i potrošnje reaktivne energije, a sastoji se od nadnaponske zaštite (U>) koju čine trofazni naponski relej najmanjeg opsega podešenja (0,9-1,2) Un, koja reaguje sa vremenskom zadrškom najmanjeg opsega podešavanja (0,2-3 s) i podnaponske zaštite (U<) koju čini trofazni naponski relej najmanjeg opsega podešavanja (1,0 – 0,7) Un, koja reaguje sa vremenskom zadrškom najmanjeg opsega podešavanja (0,2-3 s). Frekventna zaštita koja reaguje na poremećaj ravnoteže između proizvodnje i potrošnje aktivne energije,a sastoji se od nadfrekventne i podfrekventne zaštite. Nadfrekventna zaštita (f>) koju čini monofazni frekventni relej najmanjeg opsega podešavanja (49-52 Hz), koja reaguje sa vremenskom zadrškom najmanjeg opsega podešavanja (0,2-3 s). Podfrekventna zaštita (f<) koju čini monofazni frekventni relej najmanjeg opsega podešavanja (51-48 Hz), koja reaguje sa vremenskom zadrškom najmanjeg opsega podešavanja (0,2-3 s), a frekventni relej treba da bude sa funkcijom brzine promene frekvencije u intervalu 10 mHz.

Zaštita 20 kV priključnog voda koji se ugrađuje na strani elektrane čini: prekostrujna zaštita, koja je trofazna, maksimalna strujna vremenski nezavisna zaštita, koja reaguje :

- sa vremenskom zadrškom najmanjeg opsega podešavanja (0,2-3 s), pri strujnim opterećenjima koja prelaze vrednosti dozvoljenih stujnih opterećenja priključnog voda – prekostrujna zaštita I >,
- trenutno pri bliskim kratkim spojevima – kratkospojna zaštita I>>.

Merni releji prekostrujne zaštite su za naznačenu struju 5 A i najmanji opseg podešavanja :

- (3-9) A za prekostrujnu zaštitu I >,
- (20-50) A za kratkospojnu zaštitu I >>.

Zaštita transformatora 20/ 0,4 kV elektrane na biogas od unutrašnjih kvarova je predviđena PT100 senzorom

temperature koji deluje na isključenje sklopke rastavljača u ćeliji SN razvodnog postrojenja. Zaštita transformatora od preopterećenja je obezbeđena sa temičkim relejom, dok je kratkospojna zaštita obezbeđena sa mikroprocesorskim zaštitnim uređajem koji deluje na prekidač na prekidačkom bloku SN. Nema opasnih prenapona jer visoki objekat ima gromobransku instalaciju, čiji prihvatni sistem se preko spusnih provodnika i zemljovoda direktno priključuje na temeljni uzemljivač.

Termičku zaštitu čine termički relej (digitalni termometar) i po jedan otporni senzor (sonda) Pt 100 za svaki namotaj. Oni omogućavaju praćenje kontinualno temperature najtoplije tačke svakog namotaja i signalizacija se vrši ako neki od namotaja dostigne temperaturu θs=140 °C, dok se komanda za isključenje transformatora daje kada temperatura namotaja dostigne vrednost θi = 150°C.

U slučaju reagovanja bilo sistemske zaštite ili zaštite priključnog voda spojni prekidač vrši automatsko prekidanje paralelnog rada elektrane sa distributivnim sistemom električne energije. Orman rasklopne opreme sastoji se od prekidačkog i transformatorskog polja i dat je na slici 3.

Slika 3. *Spoljašnji izgled ormana prekidačke/transformatorske ćelije*

Spojni prekidač koji se nalazi u prekidačkom polju je opremljen relejom koji poseduje sve neophodne zaštite za bezbedan paralelan rad elektrane i njegov izgled je dat na slici 4.

Slika 4. *Spoljašnji izgled prekidačke/transformatorske ćelije*

Analizom signala sa strujnih i naponskih mernih transformatora koji se nalaze u istoj prekidačkoj ćeliji, dolazi do reagovanja releja, odnosno spojnog prekidača. Kao što može da se vidi sa slike 4, crvena lampica na releju pokazuje stanje releja odnosno da li je zaštita reagovala. U normalnom pogonu pokazuje pristustvo napona od 20 kV. Od zaštita na releju su prikazane prekostrujna zaštita, kratkospojna zaštita, prekostrujna zaštita sa vremenskim kašnjenjem, nadnaponska, podnaponska, nadfrekventna, podfrekventna, ROCOF (rate of change of frequency) zaštita, ako dođe do gubitka SF6 gasa zaštita i ako dođe do velikih ispada automatsko isključenje. ROCOF predstavlja veliku promenu u frekvenciji usled ozbiljnih sistemskih incidenata. ROCOF zaštita omogućava reagovanje releja pre nego što frekvencija postane previsoka ili preniska.

6. ZAKLJUČAK

Elektrane na biogas su značajne, jedan nezamenjivi izvor energije u savremenim elektroenergetskim sistemima. U ovom radu je posebna pažnja posvećena zaštitama u elektranama na biogas u kojima se koristi mikroprocesorska zaštita. Zaštite koje se koriste su nadnaponska, podnaponska, nadfrekventna, podfrekventna, termička zaštita, zaštita od preopterećenja, trenutna prekostrujna zaštita, zemljospojna zaštita, ako dođe do gubitka SF6 gasa zaštita i automatsko isključenje ako dođe do velikih ispada. Svaka žaštita ima karakteristična podešenja koja omogućavaju pouzdanost i sigurnost elemenata koje štite. Delovanjem ovih zaštita mora se na spojnom prekidaču izvršiti automatsko prekidanje paralelnog rada elektrane sa distributivnog sistema električne energije da ne bi došlo do neželjenih dejstva.

7. LITERATURA

[1] Enrique Alberto Huerta-Reynoso, Jorge Alberto Gómez: *Biogas Power Energy Production from a Life Cycle Thinking*, 2018.

[2] Nikola Lazić: *Biogas i njegova primena (master rad)*, Fakultet tehničkih nauka, Novi Sad, 2010.

[3] Duško Bekut: *Relejna zaštita*; Fakultet tehničkih nauka, Novi Sad, 2009.

[4] G. Dotlić : *Elektroenergetika – kroz standarde, zakone, pravilnike, odluke i tehničke preporuke*, 2004

Kratka biografija:

Stefan Stepanović rođen je u Novom Sadu 1995. godine. Osnovne studije završio je na Fakultetu tehničkih nauka 2019. godine iz oblasti Elektrotehnika i računarstvo smer Elektroenergetski sistemi. Master rad, na istom fakultetu, na smeru Elektroenergetski sistemi je odbranio 2021. godine.

Kontakt: stefanjuve17@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/15BE21Bosancic

REGULACIJA NAPONA KLASIČNE DISTRIBUTIVNE MREŽE SA DISTRIBUTIVNIM GENERATOROM

VOLTAGE REGULATION OF A CLASSIC DISTRIBUTION NETWORK WITH A DISTRIBUTION GENERATOR

Strahinja Bosančić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je analiziran uticaj distributivnih generatora (DG) na regulaciju napona u distributivnoj mreži (DM). Iznete su teorijske osnove iz aspekta regulacije napona, tokova snaga i uticaja DG na regulaciju napona. Izložena su i upoređena rešenja dobijena korišćenjem klasične i modifikovane regulacije napona. Upoređeni su rezultati dobijeni kontinualnom i diskretnom regulacijom*

Ključne reči: *Regulacija napona, distributivni generatori.*

Abstract – *The paper analyzes the influence of distribution generators (DG) on voltage regulation in the distribution network (DN). Theoretical bases from the aspect of voltage regulation, power flows and the influence of DG on voltage regulation are presented. The solutions obtained by using classical and modified voltage regulation are presented and compared. The results obtained by continuous and discrete regulation were compared.*

Keywords: *Voltage regulation, distribution generators.*

1 UVOD

Napon i učestanost elektroenergetskog sistema (EES) predstavljaju dve osnovne promenljive koje karakterišu sisteme za proizvodnju, prenos i distribuciju električne energije [1]. Prisustvo DG u DM ima veliki uticaj na struje kratkih spojeva, tokove snaga i kvalitet električne energije.

Nakon Uvoda u drugom delu rada opisan je postupak proračun tokova snaga i problem regulacije napona DM, koja se zasniva automatskom regulatoru napona, sa na zakonom regulacije. U trećem delu izvršena je postavka problema i parametara, kao i opis mreže. Zavisnost štete od pozicije regulacione sklopke za različite režime mreže, i snage DG, analizirana je četvrtom delu rada. U isto delu, upoređeni su rezultati dobijeni primenom diskretne i kontinualne regulacije napona. Zaključci rada i referentno navedena literatura koja je korišćena pri izradi ovog rada, dati su u petom i šestom delu rada, respektivno. Glava 7 sadrži prilog u kome se nalaze rezultati tokova snaga koji su primenjeni u radu. Tabelarno su prikazane vrednosti struja, napona, proizvodnje i potrošnje po granama.

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

2 OSNOVNI POJMOVI

U ovom delu prikazani su regulacioni transformatori i postupak za proračun tokova snaga, korišćeni u izradi ovog rada.

2.1 Regulacioni transformatori

Energetski transformatori su statički uređaji koji na principu elektromagnetne indukcije pretvaraju napon/struju između dva, ili više namotaja, pri istoj učestanosti, na druge vrednosti napona/struja. Ako je prenosni odnos transformatora promenljiv, onda su to regulacioni transformatori [9]. U zavisnosti da li se vrednost prenosnog odnosa menja pod opterećenjem, ili ne, oni se dele na:

- regulacione transformatore sa regulacijom pod opterećenjem (RTrPO),
- reguacione transformatore s regulacijom u beznaponskom stanju (RTrBS).

2.2 Tokovi snaga

Proračun tokova snaga se sastoji u proračunu promenljivih stanja (odnosno kompletnog režima) DM, na bazi poznatog napona izvora napajanja mreže (korena) i poznatih potrošnji u svim čvorovima mreže [6].

2.2.1 Metodologija za tokove snaga

Za razliku od prenosnih mreža, distributivne mreže su dominantno radijalne, sa veoma malim brojem petlji. Zbog te činjenice poznati i široko korišćeni iterativni algoritmi, za proračune tokova snaga u prenosnim mrežama, zasnovanih na matričnim modelima, nisu dovoljno efikasni za proračune tokova snaga u DM. Razvojem specijalizovanih algoritama za proračune tokova snaga u DM [6], koji se zasnivaju na karakteristikama DM (pre svega da je DM dominantno radijalna mreža), prethodno navedeni problemi su prevaziđeni.

2.2.2 Algoritam sumiranja struja

Jedna od najčešće korišćenih specijalizovanih algoritama za proračune tokova snaga u DM, Algoritam sumiranja struja, poznatija kao Shirmohammadi-ev algoritam [3], predstavlja iterativni postupak za proračun tokova snaga. Algoritam počinje inicijalizacijom postupka, a nakon toga sledi iterativni postupak, koji se sastoji od tri koraka:

1. Proračun injektiranih struja prema relaciji:

$$i_i^{(h)} = \left(\frac{s_{p,i}}{V_i^{(h-1)}}\right)^* + y_{0,i} \cdot V_i^{(h-1)}, i = 1, \dots, n_{cv}.$$

2. Proračun struja po granama prema relaciji:

$$j_i^{(h)} = i_i^{(h)} + \sum_{j \in i} j_j^{(h)}, i = n_{gr}, \dots, 1.$$

3. Proračun napona u čvorovima prema relaciji:

$$V_i^h = V_{i-1}^{(h)} - z_i \cdot j_i^{(h)}, i = 1, \dots, n_{cv}.$$

3 POSTAVKA PROBLEMA

U ovom delu razmatra se uticaj DG na DM kao i postupak formiranja zakona regulacije.

3.1 Uticaj DG na DM

Primena DG u distributivnom sistemu ima brojne pogodnosti. Sa ekonomskog stanovišta to su [8]: pokrivanje povećanja potrošnje određenog broja potrošača, mogućnost proširenja DG u malim koracima, upravljanje potrošnjom. Sa stanovišta eksploatacije sistema [8]: pozitivan uticaj na naponske prilike, smanjenje gubitaka energije, poboljšanje pouzdanosti.

Pored prethodno navedenih pozitivnih efekata, priključenje DG u DM ima i neke negativne posledice. Tako na primer, priključenjem DG, u DM koje su projektovane kao pasivne mreže (tokovi aktivne snage su uvek od mreže višeg ka mreži nižeg napona), moguće je da tokovi aktivnih snaga imaju oba smera. To dovodi do problema vezano za stabilnost, upravljanje i zaštitu mreže.

3.2 Formiranje zakona regulacije

Zakon regulacije se formira za DM bez uticaja DG. Optimalna pozicija ragulacione sklopke se računa za tri različite vrednosti potrošnje. Zatim se navedeni postupak ponavlja, ali sa uvaženim DG. Razmatra se i situacija kada se za proračun štete uvažavaju i potrošači na tercijeru. U tom slučaju, vrednosti optimalne pozicije regulacione sklopke su veće u odnosu na pozicije dobijene sa zanemarenim potrošačima na tercijeru.

4 VERIFIKACIJA PROBLEMA REGULACIJE NAPONA DM SA DG

U ovoj glavi obrađeno je više različitih problema regulacije napona DM sa DG. Analizirana je promena štete u zavisnosti od pozicije regulacione sklopke za različite vrednosti proizvodnje i potrošnje. Upoređeni su rezultati dobijeni kontinualnom i diskretnom regulacijom. Takođe, analiziran je uticaj promene snage DG na štetu, struju, napon i gubitke u mreži.

4.1 Test distributivna mreža

Test distributivna mreža prikazana je na slici 4.1. Mreža se sastoji od jednog trofaznog tronamotajnog regulacionog transformatora (Tr), sa pet različitih izvoda na sekundaru i pet različitih izvoda na tercijeru. Svi izvodi su jednakih dužina, 10 km:

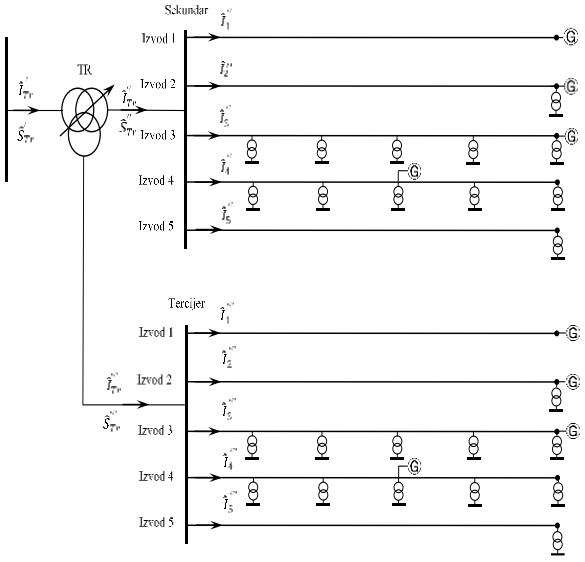
I izvod - nema potrošače, na kraju je priključen DG;

II izvod - na kraju je priključen potrošač i DG;

III izvod - ima N jednakih, ravnomerno raspoređenih potrošača, kao i DG na kraju voda;

IV izvod - ima 5 jednakih ravnomerno raspoređenih potrošača i DG na sredini voda;

V izvod - nema DG, na kraju je priključen potrošač.

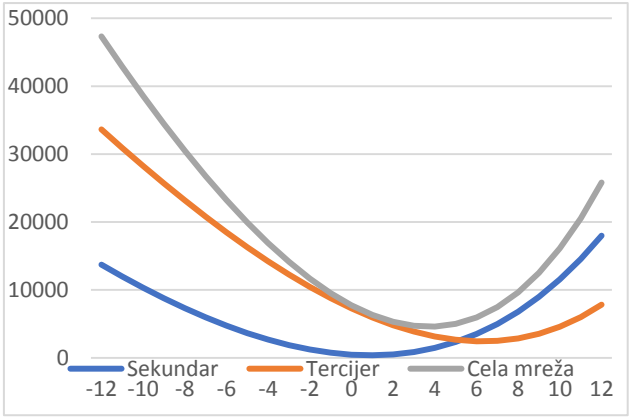


Slika 4.1 – Test distributivna mreža

4.2 Zavisnost štete od promene pozicije regulacione sklopke

U ovom poglavlju analizirana je promena vrednosti štete koju potrošači trpe usled odstupanja napona od nominalne vrednosti, u zavisnosti od snage DG. Snaga koju DG injektiraju u mrežu menja vrednost u rasponu od 0 MW do 8 MW. Pritom, vrednosti ukupne snage potrošnje na svakom od izvoda sa potrošačima imaju jednu od sledeće tri vrednosti: (4+j2) MVA, (4–j2) MVA i (4+j0) MVA.

Zavisnost štete od pozicije regulacione sklopke za slučaj, u kome je snaga potrošača S_p=(4+j0) MVA, a proizvodnja DG S_G=0 MW, prikazana je na slici 4.2.



Slika 4.2 – Zavisnost štete od pozicije regulacione sklopke

Sa slike se može uočiti sledeće, kad se posmatraju:

- samo potrošači koji se napajaju preko sabirnica sekundara (plava linija), optimalna pozicija regulacione sklopke je 1, a vrednost štete 372.19 n.j.;
- samo potrošači sa tercijera (narandžasta linija), optimalna pozicija regulacione sklopke je 6, a vrednost štete 2438.03 n.j.;
- svi potrošači (siva linija) optimalna pozicija regulacione sklopke je 4, a vrednost štete 4606.04 n.j.

Očekivano, optimalna pozicija regulacione sklopke je veća kada se posmatraju samo potrošači na tercijeru (10 kV), u odnosu na poziciju kada se posmatraju samo potrošači na sekundaru (20 kV). Komentar: za isto opterećenje, na tercijeru, gde je niži napon, veće su struje, pa samim tim je veći pad napona.

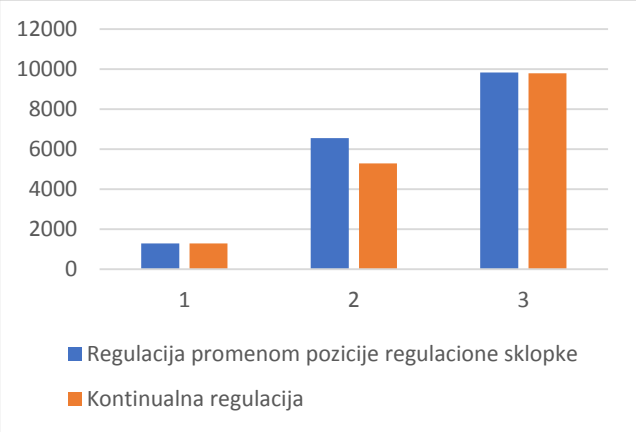
4.3 Kontinualna regulacija

Minimalne vrednosti štete mogu se proračunati pomoću dve različite metode [9]:

- regulacija promenom pozicije regulacione sklopke (diskretna regulacija),
- kontinualna regulacija.

Na slici 4.3 prikazane su vrednosti minimalne štete u zavisnosti od tipa regulacije. Minimalna vrednost štete dobijena promenom pozicije regulacione sklopke je naznačena plavom bojom, a minimalna vrednost štete dobijena primenom kontinualne regulacije je naznačena narandžastom bojom. Grafik se sastoji od tri para vrednosti, vrednost štete kada se:

1. uvažavaju samo potrošači sa sekundara,
2. uzima u obzir samo potrošače sa tercijera,
3. obuhvataju svi potrošače u mreži.



Slika 4.3 – Minimalna šteta u zavisnosti od tipa regulacije

Prilikom određivanja minimalne štete promenom pozicije regulacione sklopke, napon korena je konstantan i iznosi 110 kV. Kod kontinualne regulacije, postupak je drugačiji. Pozicija regulacione sklopke se postavi na nulu i ne menja se tokom proračuna. U tom slučaju napon se menja u opsegu od 90 kV do 130 kV sa korakom 0.2 kV.

U tabeli 4.3, prikazane su vrednosti za sledeći primer: ukupna snaga na izvodu je S_P= (4+j2) MVA, a snaga DG je S_G=0 MW.

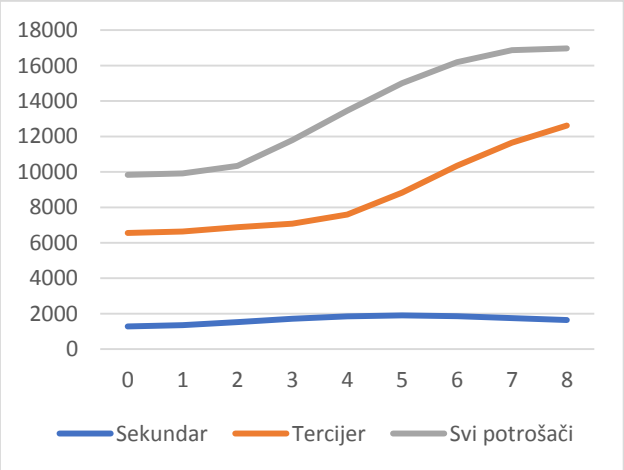
Tabela 4.3 – Vrednosti štete pri različitim regulacijama

Diskretna regulacija			Kontinualna regulacija		
Napon [kV]	Opt. poz.	Šteta [n.j]	Napon [kV]	Opt. poz.	Šteta [n.j]
110	8	1281.20	120.2	0	1281.20
110	12	6551.60	130.0	0	5285.99
110	12	9834.93	125.6	0	9795.54

Sa grafika se vidi da su vrednosti štete dobijene diskretnom regulacijom uvek veće od vrednosti dobijenih kontinualnom regulacijom. To se dešava zbog toga što se minimalna vrednost štete dobijena kontinualnom regulacijom određuje tako što se napon korena menja u veoma malim koracima. Kako su promene napona na sekundaru i tercijeru dosta manje, postižu se bolje vrednosti napona, a samim tim je i šteta je manja.

4.4 Minimalna šteta u zavisnosti od promene snage DG

Razmatra se zavisnost štete od promene snage DG, pri čemu su pozicije regulacione sklopke fiksirane na prethodno određenim optimalnim vrednostima – vrednosti koje su prikazane u delu 4.2. Snaga DG se menja od 0 MW do 8 MW. Na slici 4.4, prikazana je zavisnost štete od promene snage DG (y osa - šteta, x osa - snaga DG.). Plavom bojom je prikazana zavisnost štete od snage DG za sve potrošače koji se napajaju preko sekundara Tr. Narandžastom bojom uvaženi su samo potrošači koji se napajaju preko tercijera Tr, dok je sivom bojom prikazana ukupna šteta, koja obuhvata sve potrošače razmatrane mreže.



Slika 4.4 – Zavisnost štete od promene snage DG

U tabeli 4.4 prikazani se su vrednosti za sledeći primer: ukupna snaga potrošača na izvodu je S_P=(4+j2) MVA, a optimalne pozicije regulacione sklopke za potrošače na sekundaru, tercijeru i sve potrošače DM su 8, 12 i 12, respektivno. Zelenom bojom prikazane su minimalne, a ljubičastom maksimalne vrednosti štete.

Tabela 4.4 – Šteta u zavisnosti snage DG

S_G [MW]	Primar [n.j]	Sekundar [n.j]	Svi [n.j]
0	1281.20	6551.60	9834.93
1	1351.50	6636.37	9911.16
2	1528.82	6880.94	10335.23
3	1717.43	7082.34	11791.26
4	1852.56	7595.79	13458.84
5	1899.89	8830.00	15007.43
6	1855.92	10344.92	16194.63
7	1749.46	11651.93	16868.13
8	1644.34	12617.01	16972.38

Sa grafika se može uočiti da su vrednosti štete na tercijeru

znatno veće nego vrednosti štete na sekundaru. To se dešava zbog toga što su svi parametri i potrošnja na sekundaru i tercijeru isti, ali je struja na tercijeru duplo veća. Šteta je najveća kada se posmatra čitava mreža.

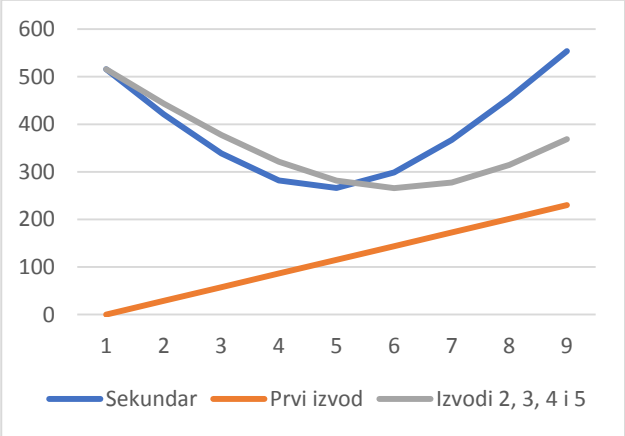
4.5 Uticaj promene snage DG na struje, napone i gubitke u mreži

Razmatra se zavisnost struje, napona i gubitaka snage od snage DG koja se menja u opsegu od 0 MW do 8 MW. U prikazanim primerima, ukupna snaga potrošača u čitavoj DM je (4+j2) MVA. Dobijeni rezultati prikazani su u tabeli 4.5. Zelena polja u tabeli označavaju minimalne, a ljubičasta maksimalne vrednosti struja.

Tabela 4.5 – Zavisnost struje od snage generatora

S_g [MW]	Struja na sekundaru Tr [A]	Struja na prvom izvodu [A]	Suma struja sa izvoda 2, 3, 4 i 5 [A]
0	515.88	0	515.88
1	420.82	28.75	443.57
2	338.87	57.5	377.46
3	281.81	86.25	321.46
4	266.24	115	281.73
5	298.8	143.75	265.73
6	366.99	172.5	277.67
7	455.14	201.25	314.48
8	553.88	230	368.87

Na slici 4.5 prikazana je zavisnost struje od snage DG (x osa - snaga DG, y osa - struja).



Slika 4.5 – Zavisnost struje od snage DG

Narandžastom linijom na grafiku prikazana je promena struje na prvom izvodu u zavisnosti od snage DG. Sa povećanjem snage DG, povećava se i struja. Povećanje struje je linearno, jer se na prvom izvodu nalazi samo DG.

Struja na sekundaru transformatora je prikazana plavom bojom. Kada je snaga DG 0 MW, nema proizvodnje i tada sva struja prelazi preko transformatora. Sa povećanjem snage DG, struja na sekundaru transformatora se smanjuje, jer sada deo snage dolazi i iz mreže. Struja se smanjuje do snage DG od 4 MW. Tada su proizvodnja i potrošnja aktivne snage u mreži iste i zbog toga struja ima minimalnu vrednost koja iznosi 266.24 A. Nakon toga, sa povećanjem proizvodnje, javlja se višak aktivne snage u DM, pa se ta snaga vraća u prenosnu mrežu. Samim tim se struja na sekundaru transformatora povećava. Maksimalna vrednost iznosi 553.88 A i dostiže se za snagu DG od 8 MW.

Sivom linijom prikazana je ukupna struja na izvodima 2, 3, 4 i 5. Sa povećanjem proizvodnje DG, smanjuje se struja na izvodima. Njena vrednost se smanjuje sve do snage DG od 5 MW, kada dostiže minimalnu vrednost 265.73 A. Nakon toga ponovo počinje da se povećava. Kriva ima sličan oblik kao u drugom slučaju.

5 ZAKLJUČAK

Priključenje DG na sistem drastično menja prirodu postojeće radijalno napajane distributivne mreže, od pasivne u aktivnu. Priključenje DG u manjoj ili većoj meri, menjaju se tokovi snaga, utiče se na rad zaštitnih uređaja, nivo struja kratkog spoja, stabilnost sistema, kvaliteta regulacije napona, vrednost gubitaka snage u mreži, kvalitet električne energije, itd. U ovom radu je razmatran uticaj DG na rad klasične regulacije napona, u kojoj je zakon regulacije formiran bez prisustva DG.

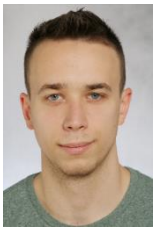
U skladu sa tim, priključenjem DG, regulacioni transformator ima lažnu sliku o ukupnom opterećenju svih potrošača koji se preko njega napajaju. Odnosno, priključenjem DG smanjuje se vrednost optimalne pozicije regulacione sklopke, i potrošačima se isporučuje električne energija sa naponom čija je vrednost ispod vrednosti koja je definisana zakonom regulacije.

Pored uticaja DG, na zakon regulacije veoma veliki uticaj ima činjenica da li se za određivanje minimalne štete posmatraju samo potrošači na sekundarnoj strani transformatora, ili se uvažavaju i potrošači napajani sa tercijera.

6 LITERATURA

1. G.Švenda: *Specijalizovani softveri u elektroenergetici*, skripta sa predavanja iz istoimenog predmeta na master studijama, Fakultet Tehničkih Nauka, Novi Sad, 2019.
2. M.Jajčanin: *Izbor optimalnih pozicija regulatora napona u radijalnim distributivnim mrežama*, Master rad, Fakultet Tehničkih Nauka, Novi Sad, 2016.
3. V. Strezoski: *Sistem regulacije napona distributivnih mreža*, Fakultet Tehničkih Nauka, Novi Sad, 1997.
4. G. Švenda: *Osnovi elektroenergetike – Matematički modeli i proračuni*; edicija: Tehničke nauke – Udžbenici, br. 177, FTN, Novi Sad, Novi Sad, 2007.

Kratka biografija:



Strahinja Bosančić rođen je u Novom Sadu 1995. godine. Diplomski rad na Fakultetu tehničkih nauka u Novom Sadu odbranio je 2019. godine u oblasti Elektrotehnika i računarstvo. Odmah po završetku osnovnih studija, upisuje master na smeru Elektroenergetski sistemi.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.39
DOI: https://doi.org/10.24867/15BE22Todic

DETEKCIJA NASELJA NA SATELITSKIM SNIMCIMA

DETECTION OF SETTLEMENTS IN SATELLITE IMAGES

Tamara Todić, *Fakultet tehničkih nauka, Novi Sad*

Oblast - ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj - *U radu je razmatran problem klasifikacije sadržaja multitemporalnih, multimodalnih i multispektralnih slika dobijenih pomoću sistema daljinske detekcije, a u cilju pronalaženja naselja. Akcenat je na integrisanju podataka iz različitih izvora.*

Ključne reči: *detekcija naselja, satelitski snimci, neuronska mreža, fuzija podataka*

Abstract - *The paper considers the problem of classification of content of multitemporal, multimodal and multispectral remote sensing images, with the aim of settlement detection. The accent is on the integration of data from multiple different sources.*

Keywords: *Settlement detection, satellite images, neural network, data fusion*

1. UVOD

Sistemi za opservaciju Zemlje svakodnevno generišu ogromne količine podataka, koji se mogu koristiti u razne svrhe, između ostalog, za detekciju naseljenih mesta, što je i tema ovog rada.

U 2012. godini postignut je značajan napredak što se kvaliteta satelitskih snimaka tiče - lansirani su novi sateliti sa opremom koja je omogućila da se dobiju slike veće rezolucije, tačnosti i osetljivosti. Posebno su zanimljivi snimci dobijeni noću - jer se tada uočavaju osvetljeni delovi planete, što je važno za brojna istraživanja - između ostalog, i za temu kojom se bavi ovaj rad - detekcija naselja [1].

Za detekciju naselja, informacija o tome gde su osvetljene oblasti je veoma važna. Ali, naselja koja nemaju pristup električnoj energiji ne mogu biti detektovana samo na osnovu prisustva svetlosti tokom noći. Za taj problem potrebno je iskoristiti i dnevne snimke. Zatim, korisno je imati snimke iste oblasti kroz vreme, jer se oblasti sa gustom vegetacijom menjaju u skladu sa godišnjim dobima, za razliku od urbanih oblasti.

2. PREGLED LITERATURE

U radu [2], integrišu se vremenske i prostorne informacije u cilju klasifikacije piksela satelitskih slika sa SENTINEL-2 i VHSR u jednu od 12 klasa. Koristi se model koji ima 2 grane, jednu koja obrađuje vremenski niz podataka, i drugu koja je zadužena za prostorne informacije. Vrši se fuzija obeležja dobijenih iz ove dve grane, i nad tim se radi klasifikacija, a takođe se klasifikuju i izlazi svake grane zasebno, kako bi se mreža podstakla da izvuče maksimalnu količinu informacija iz svake vrste podataka. Za poređenje rezultata korišćen je *Random Forest* (RF), i performanse M^{3}Fusion modela prevazilaze performanse *Random Forest* algoritma. Rezultati za pet različitih slučajnih trening-test podela podataka, dobijeni na osnovu snimaka francuskog ostrva Reunion, iz 2016. i 2017. godine, dati su u tabeli 1.

Tabela 1. Rezultati *M^{3}Fusion* modela za klasifikaciju piksela sa satelitskih snimaka u poređenju sa RF, prema [2]

br. eksp.	RF		M^{3}Fusion		Gain	
	Acc.	F-Meas	Acc.	F-Meas	Acc.	F-Meas
1	87,39	87,11	90,67	90,67	+3,28	+3,56
2	88,47	88,05	91,52	91,39	+3,05	+3,34
3	85,21	84,62	89,25	89,15	+4,04	+4,53
4	88,33	88,05	90,61	90,7	+2,28	+3,08
5	87,29	86,88	90,09	89,96	+2,8	+3,08

U radu [3] koriste se konvolucione neuronske mreže za klasifikaciju urbanog otiska - najpre su korišćene poznate arhitekture poput U-Net, SegNet i FCN, ali samo na slikama iz jednog trenutka - ne na vremenskim nizovima, a zatim je arhitektura koja je pokazala najbolji rezultat, u ovom slučaju U-Net, trenirana na vremenskim nizovima snimaka. Kada se koristi vremenski niz snimaka, performanse se popravljaju za 2.72% i 6.4% za dve klase koje postoje u odnosu na slučaj kad se koristi samo slika iz jednog trenutka.

U radu [4] koristi se konvoluciona neuronska mreža kako bi se detektovala neformalna naselja na VHR (engl. Very High Resolution) satelitskim snimcima. Pošto je potrebno klasifikovati svaki piksel, tokom treninga, u mrežu ulaze delovi slike, engl. *patch*, koji okružuju piksel od interesa i model se trenira da izvrši klasifikaciju jednog *patch*-a. Kada se dobije istreniran model, on se koristi tako što klizajući prozor veličine *patch*-a iz treninga prelazi preko slike pri čemu se pomera za po jedan piksel, i na svakom mestu se vrši klasifikacija *patch*-a. Postignuta je tačnost 91,71%.

3. OPIS BAZE PODATAKA

U ovom istraživanju korišćena je baza podataka koja je data na *2021 IEEE GRSS Data Fusion Contest: Track DSE* takmičenju, gde je cilj bio da se uz pomoć multitemporalnih, multimodalnih i multispektralnih satelitskih slika detektuju naselja koja nisu snabdevena električnom energijom. Takmičenje je organizovano od strane odbora

---

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Branko Brkljač, docent.

Image Analysis and Data Fusion Technical Committee (IADF TC), u okviru udruženja *IEEE Geoscience and Remote Sensing Society* (GRSS), sa ciljem da se promoviše istraživanje u oblasti daljinske detekcije u svrhe automatskog pronalaženja naselja bez pristupa električnoj energiji.

Bazu podataka čini 98 slika veličine 800x800 piksela, rezolucije 10m. Ravnomerno su raspoređene u trening, validacioni i test skup (60/19/19). Test skup nije vidljiv takmičarima, a za validacioni nisu dostupne labele [5].

Svaka slika ima 98 kanala, i sve slike su skalirane tako da GSD (engl. Ground Sampling Distance) bude 10m. Svaka slika dakle ima površinu 64km^2. Snimci su dobijeni sa četiri satelita: SENTINEL-1 - vremenski niz od 4 snimka po 2 kanala koja odgovaraju VV i VH polarizaciji; SENTINEL-2 - vremenski niz od 4 snimka po 12 kanala iz vidljivog i SWIR (engl. *Short Wave Infrared*) dela spektra; Landsat 8 - vremenski niz od 3 snimka po 11 kanala iz VNIR (engl. *Visible Near InfraRed*), SWIR i TIR (engl. *Thermal InfraRed*) delova spektra, s tim da je izostavljen opseg talasnih dužina koje se koriste za detekciju cirusa (oblaka), jer ne sadrži informacije o tlu; VIIRS - vremenski niz od 9 snimaka po 1 kanal koji sadrži noćne sat. snimke.

Za svaku sliku date su po dve labele/oznake - jedna u formi indeksirane slike veličine 800x800 piksela, prostorne rezolucije 10m, na kojoj svaki region veličine 500x500m (zona od 50x50 piksela) ima jedinstvenu klasu (oznaku); i druga labela u formi slike veličine 16x16 piksela, gde svaki piksel ima svoju klasu, a prostorna rezolucija pojedinačnog piksela je 500m. Ukupno postoje 4 klase ili kategorije naselja: 1) bez električne energije, 2) naselja sa električnom energijom, 3) region bez naselja ali sa električnom energijom, i 4) region bez naselja i bez električne energije. Odgovarajuće numeričke oznake date su u tabeli 2.

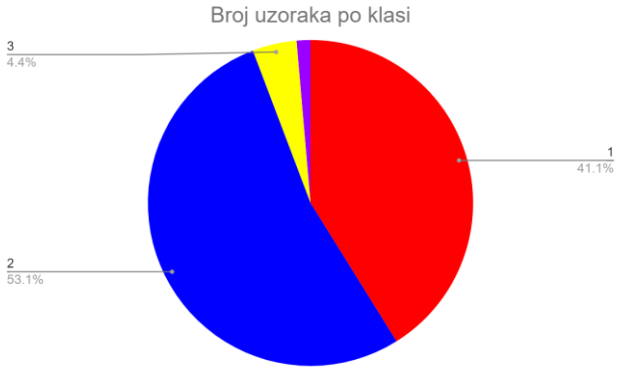
Tabela 2. Vrednosti piksela i boje koje odgovaraju klasama u korišćenom skupu satelitskih snimaka [5]

Vrednost piksela	Klasa	Boja
1	Naselje bez električne energije (klasa od interesa)	ff0000
2	Bez naselja i bez električne energije	0000ff
3	Naselja sa električnom energijom	ffff00
4	Bez naselja sa električnom energijom	b266ff

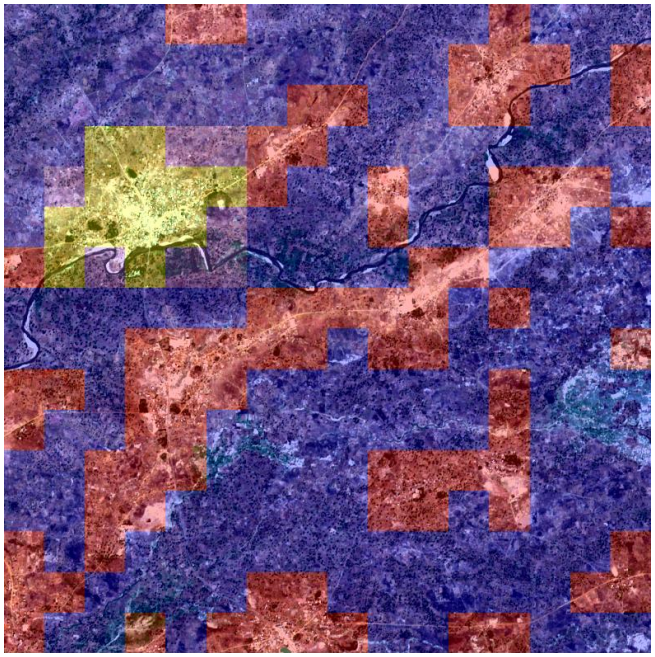
Zastupljenost pojedinih klasa u skupu podataka prikazana je na slici 1, a primer kako su labelirane slike na slici 2.

Skup podataka nije izbalansiran, najveći deo (53,1%) čine uzorci iz klase bez naselja i bez električne energije, 41,1% su uzorci iz klase naselja bez električne energije, dok klasa naselja sa električnom energijom čini 4,4% skupa, a klasa bez naselja, sa električnom energijom samo 1,4% [5].

Slika 3 prikazuje broj slika iz skupa podataka koje na sebi sadrže svaku od klasa. Oznake i boje odgovaraju onima u tabeli 2.



Slika1. Raspodela klasa u skupu podataka



Slika 2. SENTINEL-2 slika sa labelama [5]

Na svakoj slici u skupu podataka za trening javljaju se klase naselja bez električne energije i bez naselja i bez električne energije. Klasa naselja sa električnom energijom javlja se na 38% trening slika, a region bez naselja sa električnom energijom na 32% trening slika.

Slika 3. Broj slika iz skupa podataka koje na sebi sadrže određenu klasu.

Svaka slika koja sadrži region bez neselja sa električnom energijom sadrži i sve ostale klase. 19 od 60 slika iz trening skupa sadrži sve 4 klase, dok 37 slika sadrži samo

klase naselja bez električne energije i bez naselja i električne energije (klase 1 i 2).

4. MODEL

Kako su u bazi dostupni vremenski nizovi multispektralnih slika sa različitih satelita, ideja je da se model za detekciju naselja projektuje tako da koristi sve raspoložive podatke. Po ugledu na M^{3}Fusion model [2], razvijena je neuronska mreža koja klasifikaciju dela slike (engl. *patch*-a) realizuje kroz fuziju podataka i sastoji se iz dve grane za obradu.

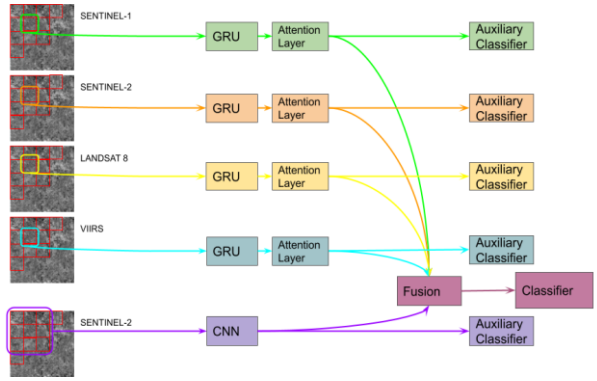
Jedna grana ima fokus na podacima kroz vreme, i sadrži rekurentnu neuronsku mrežu (RNN), tačnije GRU (engl. *Gated Recurrent Unit*). Ulaz u ovu granu je vremenski niz jednog dela slike za koji je data jedinstvena labela (jedan *patch*) veličine 50x50 piksela (500x500m). Nakon GRU nalazi se "*attention*" mehanizam, sa idejom da nauči koji vremenski trenutak je koliko važan za klasifikaciju.

Druga grana je sa fokusom na prostorne informacije - ulaz je deo satelitske slike veličine 150x150 piksela (1500x1500m), koji predstavlja okolinu *patch*-a od interesa. Tu sliku obrađuje konvoluciona neuronska mreža (CNN). Ideja je da se posmatranjem okoline patcha od interesa dobijaju korisne informacije za njegovu klasifikaciju.

Izračunata obeležja na izlazu dve opisane grane se konkateniraju i zatim prolaze kroz klasifikator - potpuno povezanu neuronsku mrežu.

Dodatno, kako bi se podstaklo da mreža i iz vremenskih i iz prostornih podataka izvuče što više diskriminativnih informacija, koriste se dva pomoćna klasifikatora - jedan koji prepoznavanje kategorija realizuje samo na osnovu obeležja dobijenih iz grane koja obrađuje podatke kroz vreme, i drugi koji se oslanja samo na obeležja iz grane koja posmatra okolinu regiona od interesa [2].

Ako se koriste snimci sa više satelita u skladu sa tim se povećava i broj grana koje sadrže GRU, i za svaku se dodaje još po jedan pomoćni klasifikator. Na slici 3 je prikazana šema modela kada se koriste sva 4 satelita.



Slika 3. Šema korišćenog eksperimentalnog modela

5. EKSPERIMENTI

U cilju poređenja GRU i konvolucione GRU, kao i korišćenja svih raspoloživih podataka spram korišćenja samo nekih satelita, izvršeno je nekoliko eksperimenata.

U svakom eksperimentu, na ulazu u granu koja sadrži CNN je SENTINEL-2 slika. Poređene su performanse sistema za prepoznavanje kada se na ulaz grane koja sadrži RNN postavi samo VIIRS, zatim kombinacija SENTINEL-2 i VIIRS, i na kraju kombinacija sva 4 raspoloživa satelita.

Funkcija cene koja je korišćena za optimizaciju parametara modela predstavlja zbir funkcija cene (međuentropije) za svaki klasifikator u modelu.

Za iterativnu optimizaciju korišćen je *Adam* algoritam, a brzina učenja je postavljena na 10^{-4}. Treninzi su izvršavani na NVIDIA TITAN Xp grafičkoj kartici, a potrebna memorija za jedan model sa konvolucionom GRU koji integriše sva 4 satelita je 3,9GB. Model koji ima GRU i integriše sva 4 satelita zahteva 262MB. Trajanje jedne epohe je približno 20 minuta. Za implementaciju modela korišćena je *PyTorch* biblioteka. S obzirom na to da za validacioni deo skupa podataka nisu date labele, iz skupa za obuku koji sadrži 60 slika izdvojeno je 20% za validaciju.

Pošto su korišćene satelitske slike u nekoj meri već bile sa predobradom - izvršene su geometrijske korekcije, uklonjeni oblaci i slično, dodatno je izvršena normalizacija tako da sva obeležja budu u istom opsegu - da svaki kanal ima srednju vrednost nula i jediničnu varijansu.

S obzirom da CNN modul "posmatra" deo slike oko regiona od interesa, kako bi mogli da se koriste i ivični regioni slika se pre ulaza u ovaj modul proširuje tako što se simetrično "ogleda" na ivicama. Sem toga, u okviru korišćene baze date su i labele koje sadrže 4 klase. Pošto je primarni cilj da se detektuju naselja, bez obzira na snabdevenost električnom energijom, labele su prevedene u 2 klase - po jedna za region sa i bez naselja. Regioni koji sadrže naselja sa pristupom električnoj energiji i oni koji sadrže naselja bez pristupa električnoj energiji su spojeni u jednu klasu, a regioni koji nemaju naselja ali imaju električnu energiju i oni koji nemaju ni naselja ni električnu energiju u drugu klasu (kategoriju).

6. REZULTATI

Vrednosti funkcije cene i tačnosti na validacionom delu podataka koje su postignute u 6 eksperimenata date su u tabeli 3. Prikazana vrednost za funkciju cene izračunata je tako što je funkcija cene koja je korišćena tokom treninga (zbir međuentropija za glavni i sve pomoćne klasifikatore) podeljena sa brojem klasifikatora. Tačnost je data samo na osnovu glavnog klasifikatora.

Tabela 3. Funkcije cene i tačnost na validacionim podacima

Sateliti (korišćena merenja)	GRU		Konvoluciona GRU	
	funkc. cene / br. klas.	tačnost	funkc. cene / br. klas.	tačnost
VRIIS	0.6241	0.7523	0.6096	0.7145
S-2 VIIRS	0.7039	0.7301	0.6314	0.7454
S-1, S-2 L8, VIIRS	0.7444	0.7210	0.5276	0.7650

Ukoliko se koristi konvoluciona GRU, tačnost raste kako se koristi više satelita za klasifikaciju. Najbolju tačnost na validacionom skupu podataka postigao je model sa konvolucionom GRU koji integriše sva četiri satelita čiji se snimci nalaze u bazi podataka.

U tabeli 4 prikazane su postignute tačnosti za svaki model i svaku klasu posebno. Zelenom bojom su osenčena polja kod modela koji imaju najbolje performanse za datu klasu. Najbolje su performanse za klasu bez naselja i bez električne energije, a najlošije za klasu naseljenih mesta koja nemaju električnu energiju, sem kod modela koji koristi *GRU* i sva četiri satelita - kod njega su za ovu klasu performanse veće nego za klasu bez naselja, sa električnom energijom. Takođe, ovaj model je postigao značajno veću tačnost za klasu naselja bez eketrične energije nego modeli sa konvolucionom GRU, što ni za jednu drugu klasu nije slučaj.

Tabela 4. Tačnost svakog od modela za svaku klasu posebno

Klasa	GRU			Konvoluciona GRU		
	VIIRS	S-2 VIIRS	S-1,S-2 L8 VIIRS	VIIRS	S-2 VIIRS	S-1, S-2 L8 VIIRS
1	0.1878	0.2365	0.4637	0.1981	0.2438	0.3247
2	0.9787	0.9671	0.7752	0.9706	0.9919	0.9808
3	0.8000	0.6750	0.8125	0.7625	0.7250	0.8250
4	0.4906	0.6226	0.2830	0.4906	0.7170	0.6415

7. ZAKLJUČAK

Izvedeno je nekoliko eksperimenata u cilju poređenja GRU i konvolucione GRU, kao i korišćenja nekoliko kombinacija satelita za detekciju naselja na satelitskim snimcima. Modeli sa konvolucionom GRU pokazali su se kao bolji što se tiče tačnosti, ali zahtevaju više resursa za obuku. Različite performanse za različite klase su očekivane s obzirom na to da skup podataka nije izbalansiran.

Iako iz klase sa naseljima koja su snabdevena električnom energijom ima malo uzoraka - svega 4.4% skupa podataka, tačnost za ovu klasu je dosta visoka kod svih modela. To je takođe očekivano, pošto se razlike u odnosu na regione bez naselja vide i na dnevnim i na noćnim snimcima. Potrebno je dalje uvesti izmene u arhitekturama modela kako bi se dobili bolji rezultati, i izbalansirati skup podataka kako bi se smanjile razlike u performansama među klasama.

8. ZAHVALNICA

The authors would like to thank the IEEE GRSS Image Analysis and Data Fusion Technical Committee, Hewlett Packard Enterprise, SolarAid, and Data Science Experts for organizing the Data Fusion Contest.

9. LITERATURA

[1] C. Kyba et al., "High-Resolution Imagery of Earth at Night: New Sources, Opportunities and Challenges", *Remote Sensing*, vol. 7, no. 1, pp. 1-23, 2014. Dostupno: 10.3390/rs70100001 [Pristupljeno 30. avgusta 2021].

[2] P. Benedetti, D. Ienco, R. Gaetano, K. Ose, R. Pensa and S. Dupuy, "*M*^{3}*Fusion*: A Deep Learning Architecture for Multiscale Multimodal Multitemporal Satellite Data Fusion", *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 12, pp. 4939-4949, 2018. Dostupno: 10.1109/jstars.2018.2876357 [Pristupljeno 29. avgusta 2021].

[3] L. El Mendili, A. Puissant, M. Chougrad and I. Sebari, "Towards a Multi-Temporal Deep Learning Approach for Mapping Urban Fabric Using Sentinel 2 Images", *Remote Sensing*, vol. 12, no. 3, p. 423, 2020. Dostupno: 10.3390/rs12030423 [Pristupljeno 29. avgusta 2021].

[4] N. Mboga, C. Persello, J. Bergado and A. Stein, "Detection of informal settlements from VHR satellite images using convolutional neural networks", *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2017. Dostupno: 10.1109/igarss.2017.8128166 [Pristupljeno 29. avgusta 2021].

[5] 2021 IEEE GRSS Data Fusion Contest: www.grss-ieee.org/community/technical-committees/data-fusion".

Kratka biografija:

Tamara Todić rođena je u Kikindi 1997. godine. Osnovne akademske studije na Univerzitetu u Novom Sadu uspešno je završila 2020. god. na Fakultetu tehničkih nauka, studijski program Energetika, elektronika i telekomunikacije, smer Komunikacione tehnologije i obrada signala. Master rad iz oblasti elektrotehnike i računarstva - Obrada signala, odbranila je 2021. Godine.

Kontakt: tamaratodic97@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.85
DOI: https://doi.org/10.24867/15BE23Cvetanovic

SISTEM ZA PRAĆENJE RANJIVOSTI U SOFTVERU

SOFTWARE VULNERABILITY MONITORING SYSTEM

Vladimir Cvetanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast –ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je opisan sistem za praćenje ranjivosti u softveru. Objašnjeni su osnovni pojmovi i mehanizmi uz pomoć kojih je moguće javno identifikovanje ranjivosti u softveru. Opisan je model sistema i koraci u radu sistema. Na kraju su data zaključna razmatranja i pravci kojima bi dalji razvoj ovog alata mogao da ide.*

Ključne reči: *NVD, CVSS, CVE, ranjivost, zavisnost*

Abstract – *This paper describes a system for monitoring vulnerabilities in a software. The basic concepts and mechanisms by which it is possible to publicly identify vulnerabilities in a software are explained. The system model and steps in the operation of the system are described. Finally, there are concluding considerations and directions that the further development of this tool could be done.*

Keywords: *NVD, CVSS, CVE, vulnerability, dependency*

1. UVOD

Danas postoji velika zavisnost od veb aplikacija, u rasponu od pojedinaca do velikih organizacija. Skoro sve je uskladišteno, dostupno ili trguje na vebu. Veb aplikacije mogu biti lične veb stranice, blogovi, vesti, društvene mreže, banke, agencije, forumi, aplikacije za e-trgovinu itd. Sveprisutnost veb aplikacija u našem načinu života i u našoj ekonomiji je toliko važna da ih čini prirodnom metom za zlonamerne umove koji žele da ih eksploatišu [1].

Softver otvorenog koda revolucionisao je modernu tehnološku industriju i nigde to nije evidentnije od područja razvoja veb aplikacija. Zahvaljujući otvorenim izvornim jezicima, bibliotekama, okvirima i razvojnim alatima, programeri su u stanju da kreiraju bogate i složene veb aplikacije, obično uz delić troškova - i vremena i novca - u poređenju sa prošlim godinama [2].

Iako upotreba komponenti otvorenog koda ubrzava razvoj, to košta, jer ranjivosti otkrivene u tim bibliotekama mogu uticati na zavisne aplikacije. Sve te komponente predstavljaju zavisnosti.

Termin zavisnost je široko rasprostranjen i koristi se kada se neki softver oslanja na drugi [3].

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

Veliki broj tih zavisnosti poseduju javno poznate ranjivosti koje napadači mogu da eksploatišu. Zbog toga, potrebno je vršiti redovnu analizu tih zavisnosti kako bi se utvrdilo koje su ranjivosti prisutne i kako bi se pronašao način da se te ranjivosti uklone. Metoda putem koje se ovo ostvaruje jeste provera zavisnosti (eng. *dependency check*).
Upravo tim problemom bavi se ovaj rad. U njemu je prikazano jedno rešenje za proveru javno poznatih ranjivosti. Ono obuhvata prikupljanje svih poznatih ranjivosti i generesanje izveštaja koji pokazuje koje ranjivosti postoje i kako treba da se pristupi u rešavanju tih ranjivosti

2. MEHANIZMI ZA OKTRIVANJE RANJIVOSTI

2.1 Lista ranjivosti i izloženosti

Lista ranjivosti i izloženosti (eng. *Common Vulnerabilities and Exposures*) (CVE) je industrijski standard uobičajenih naziva za javno poznate sigurnosne ranjivosti, a organizacije su ga široko usvojile kako bi obezbedile bolju pokrivenost, lakšu interoperabilnost i poboljšanje sigurnosti [4].

CVE lista sastoji se od unosa (u zajednici ih nazivaju kao CVE identifikatori, CVE imena, CVE brojevi i CVE) koji predstavljaju jedinstvene, uobičajene identifikatore za javno poznate ranjivosti u oblasti informacione bezbednosti. Svaki CVE unos sadrži: identifikator, opis i sve relevantne reference (tj. izveštaje o ranjivosti i savete) [5].

2.2 Nacionalna baza ranjivosti

National Vulnerability Database (NVD) i njen pratilac, National Checklist Program (NCP) repozitorijum, pružaju vredan i fleksibilan skup usluga korisnicima širom sveta. NVD je osnovana 2005. godine kako bi obezbedila skladište podataka američke vlade o ranjivostima softvera i podešavanja konfiguracije, uz korišćenje otvorenih standarda kako bi se obezbedile pouzdane i interoperabilne informacije o metrici uticaja ranjivosti, metodama tehničke procene, podaci o identifikaciji IT proizvoda i reference za pomoć u sanaciji [6].

Posetioci koji pretražuju NVD bazu podataka mogu da pronađu opis mnogih detalja o bezbednosnim ranjivostima u softveru kako bi im pomogli da shvate kakve su to ranjivosti i kako im treba pristupiti. To uključuje opis CVE, koji je uglavnom dat od strane korporacije MITRE. Tada se daje slika koliko određena ranjivost može da bude opasna. Koliko je neka ranjivost opasna utvrđuje se na osnovu CVSS v2 i CVSS v3 metrike. Ova metrika definiše kako je ranjivost ocenjena

(kritična, visoka, srednja, niska), kao i detalje o tome kako bi eksploatacija ranjivosti mogla da se izvrši [7].

2.3. Sistem za ocenjivanje ranjivosti

Sistem za ocenjivanje ranjivosti (eng. *The Common Vulnerability Scoring System*) (CVSS) je specifikacija za dokumentovanje glavnih karakteristika ranjivosti i merenje potencijalnog uticaja eksploatacije ranjivosti. Motivacija za razvoj CVSS-a bila je pružanje standardizovane informacije za organizacije radi postavljanja prioriteta za ublažavanje ranjivosti [8]. CVSS se deli na tri metričke grupe: osnovna (eng. *Base*), vremenska (eng. *Temporal*) i okružene (eng. *Environmental*).

Osnovna metrička grupa predstavlja svojstva ranjivosti koja se menjaju tokom vremena, poput složenosti pristupa, vektora pristupa, stepena ugroženosti, integriteta i dostupnost sistem i zahtev za autentifikaciju sistema [9].

Vremenska metrička grupa meri svojstva ranjivosti koja se menjaju tokom vremena, kao što je postojanje zvanične zakrpe ili funkcionalni eksploatacioni kod [9].

Metrika okruženja meri svojstva ranjivosti koja su reprezentativna za okruženja korisnika, kao što su prevalencija pogođenih sistema i potencijal za gubitak. Pored toga, omogućava analitičarima da prilagode CVSS rezultat u zavisnosti od važnosti pogođene komponente za organizaciju korisnika [9].

3. PREGLED POSTOJEĆIH REŠENJA

U radu [10] opisan je sistem koji služi za pronalaženje ranjivosti u veb aplikacijama. Opisan je pristup automatizaciji skeniranja ranjivosti i testiranja bezbednosti sa rešenjem za cloud koje pruža samostalno okruženje koji ujedinjuje skenere i konfiguracije u jednom kliku [10]. Alat koji su razvili omogućuje dinamičko bezbednosno skeniranje i proveru zavisnosti bez prethodnog znanja o bezbednosti, što manuelnim testerima skraćuje količinu vremena koju inače troše na konfiguraciju i postavljanje okruženja. Rešenje objedinjuje 3 skenera bezbednosti *(Zed Attack Proxy Tool, OWASP Dependency Check, FindSecBugs Plugin*) koji su konfigurisani u distribuiranom sistemu, gde svaki skener generiše XML izveštaj. Generisani izveštaji se dalje provlače kroz proces mašinskog učenja u kom se identifikuju lažno pozitivni rezultati. Nakon toga se uklanjaju lažno pozitivni rezultati i generiše se krajnji izveštaj. Zaključak do kojeg su došli jeste da njihov sistem mnogo efikasnije identifikuje ranjivosti u odnosnu na manualne testere. U ovom sistemu cilj je da se postigne automatizacija skeniranja ranjivosti i testiranja bezbednosti, ali postoji nedostatak ukoliko želi maksimalno da se iskoristi *Zed Attack Proxy Tool* prilikom skeniranja pošto zahteva poznavanje alata i dodavanje određene dodatne konfiguracije. Sistem za praćenje ranjivosti jedino zahteva Github kredencijale kako bi pristupio repozitorijumu projekta za koji se vrši skeniranje.

Rad [11] opisuje statički pristup analizi ranjivosti. Skeniran je Maven centralni repozitorijum statičkom analizom radi otkrivanja potencijalnih softverskih grešaka. Za analizu korišćen je *FindBugs* alat koji ispituje Java bajt kod da bi otkrio brojne tipove grešaka. Skup podataka sadrži metričke rezultate koje pronalazi *FindBugs* alat za svaku verziju projekta koja je uključena u Maven repozitorijum [11]. Rezultat ovog procesa pruža skup podataka kojim svi mogu pristupiti radi daljeg istraživanja i analiziranja vezano za sigurnost u svojim sistemima. Nedostatak ovog rešenja jeste što je napravljen tako da skenira ceo Maven repozitorijum i onda se kao rezultat dobije skup podataka na osnovu kojeg je potrebno zaključiti da li se neke ranjivosti iz pronađenog skupa nalaze u projektu za koji se vrši provera, dok Sistem za praćenje ranjivosti vrši skeniranje samo zavisnosti koje se nalaze u projektu i generiše izveštaj o ranjivosti i omogućava pregled statistike, što troši manje resursa i vremena

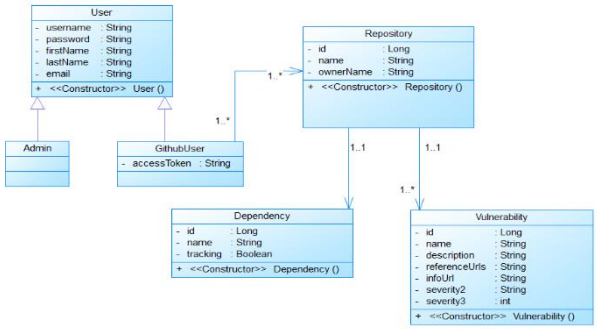
Autori rada [12] predstavljaju preciznu metodologiju koja odgovara potrebama industrije za pouzdano pronalaženje ranjivih zavisnosti u softveru otvorenog koda [12]. Ono što ova metodologija pruža jeste razlikovanje opsega zavisnosti i razmatranje direktnih i tranzitivnih zavisnosti. Pored toga, rešenje identifikuje i zavisnosti čiji je razvoj zaustavljen i uzima u obzir da zavisnosti koje se održavaju i objavljuju istovremeno predstavljaju jednu celinu prilikom analize. Za njihov rad koristili su 200 najpopularnijih biblioteka otvorenog koda koji koristi SAP *(*nem. *Systeme, Anwendugen und Produkte in der Datenverarbeitung)* u svom softveru. Zaključak do kojeg su došli jeste da se veliki procenat ranjivosti može rešiti jednostavnim ažuriranjem zavisnosti. Utvrđeno je da 20% zavisnosti koje ima ranjivost nije postavljeno (instalirano) [12], kao i da je za mali procenat zavisnosti zaustavljen dalji razvoj. Sistem za praćenje ranjivosti skenira samo direktne zavisnosti što daje dovoljno dobar izveštaj o ranjivostima koje postoje bez da troši količinu vremena koja je potrebna za opisanu metodologiju iz rada [12].

4. MODEL SISTEMA

4.1 Model sistema za praćenje ranjivosti u softveru

Ovaj sistem moguće je da koriste svi korisnici koji samostalno ili u saradnji razvijaju određeni softverski proizvod. Sistem je osmišljen tako da registrovani korisnici mogu da dodaju svoje Github repozitorijume na kojima im se nalazi projekat i za koje žele da vrše upravljanje ranjivostima. Postoji odvojeni administratorski deo aplikacije gde se dodaju korisnici kako bi mogli da pristupe ovom sistemu. Svaki registrovani korisnik ima mogućnost da doda repozitorijum, skenira repozitorijum, pregleda spisak svih pronađenih ranjivosti za skenirane repozitorijume, pregleda statistiku vezane za ranjivosti pronađene u svim repozitorijumima kao i u pojedinačnim. Pored toga, ima mogućnost da dodeli pravo pristupa za upravljanje ranjivostima drugim korisnicima kako bi oni mogli da prate ranjivosti njegovog sistema (ukoliko vrše saradnju) kao i da vrši praćenje zavisnosti koje se koriste za projekat i za koje će dobijati izveštaje. Sam model sistema za upravljanje ranjivostima i funkcionalnost skeniranja repozitorijuma opisana je u sledećem odeljku. Na slici 3 prikazan dijagram klasa opisanog sistema. U sistemu mogu da postoje dva tipa korisnika, oni su opisani sledećim klasama: *Admin* i *GithubUser*, obe klase nasleđuju klasu *User*. *GithubUser* dodatno ima polje

accessToken koje predstavlja token kojim se omogućava pristup repozitorijumu. Repozitorijum je opisan klasom *Repository* i svaki *GithubUser* može da ima jedan ili više repozitorijuma. Pored ovoga postoje klase koje opisuju zavisnost i ranjivost i to su klase *Dependency* i *Vulnerability*, respektivno. Svaki repozitorijum može da ima jednu ili više zavisnosti kao i jednu ili više ranjivost.

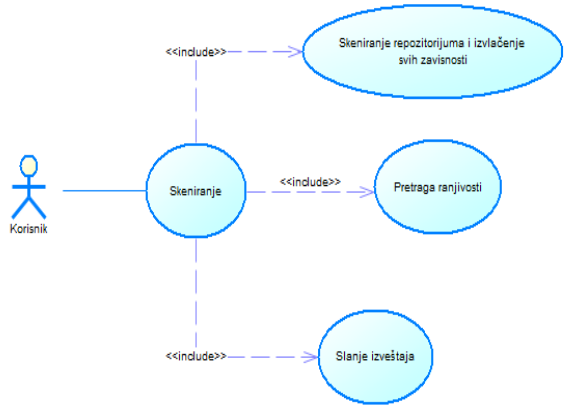


Slika 3. *Dijagram klasa sistema*

4.2 Koraci u radu sistema

Sistem za praćenje ranjivosti u softveru prilikom generisanja izveštaja o ranjivostima prolazi kroz tri faze (slika 4):

1. Skeniranje repozitorijuma i izvlačenje svih zavisnosti
2. Pretraga ranjivosti za pronađene zavisnosti
3. Slanje izveštaja



4. *Faze rada sistema*

4.2.1 Skeniranje repozitorijuma i izvlačenje svih zavisnosti

Kako bi korisnik započeo skeniranje potrebno je da izabere repozitorijum za koji želi da dobije izveštaj o ranjivostima. Ograničenje sistema se svodi na to da vrši skeniranje samo za *Maven* projekte. Nakon što korisnik izabere željeni repozitorijum, sistem vrši autentifikaciju korisnika putem *Github* API-a i pokreće skeniranje. Takođe, ovaj API omogućava sistemu da obavi sve potrebne akcije koje se izvršavaju prilikom skeniranja. Nakon što se izvrši uspešna autentifikacija, sistem parsira POM (eng. *Project Objet Model*) fajl iz projekta i iz njega izvlači listu svih zavisnosti koje se koriste. POM je osnovna jedinica rada u *Maven* projektima. To je XML datoteka koja sadrži informacije o projektu i detalje o konfiguraciji koje *Maven* koristi za izgradnju projekta [13].

4.2.2 Pretraga ranjivosti

Nakon dobijanja liste svih zavisnosti vrši se pretraga nad NVD bazom podataka i pronalaze se sve ranjivosti vezane za te zavisnosti.

Oslonac ove pretrage jeste na NVD API-u koji omogućava pretragu na osnovu ključne reči, gde ključnu reč u ovom slučaju predstavlja naziv zavisnosti.

4.2.3 Slanje izveštaja

Nakon pronalaska svih ranjivosti korisniku se šalje izveštaj na mejl i omogućava mu se da pregleda listu svih pronađenih ranjivosti na front-end delu aplikacije. Taj izveštaj sadrži sledeće informacije: naziv i link ka skeniranom repozitorijumu i listu pronađenih ranjivosti za zavisnosti koje su pronađene u repozitorijumu.

5. IMPLEMENTACIJA SISTEMA

Glavna komponenta ovog sistema jeste komponenta za skeniranje repozitorijuma. Ona se može podeliti na sledeće delove: skeniranje repozitorijuma radi pronalaženja svih zavisnosti (biblioteka), pretraga ranjivosti u NVD bazi podataka za pronađene zavisnosti i slanje mejl izveštaja korisniku za pronađene ranjivosti.

5.1 Komponenta za pronalaženje zavisnosti

Kako bi mogla da se izvrši pretraga za ranjivosti u NVD bazi, prvo je potrebno da se pronađu sve zavisnosti koje postoje unutar repozitorijuma. Metoda koja to izvršava prikazana je na listingu 1.

```
public void
searchForDependencies(RepositoryService
repoService, GitHubClient client, List<String>
dependencies, Repository repository)
        throws IOException,
ParserConfigurationException, SAXException {
    org.eclipse.egit.github.core.Repository repo
=
repoService.getRepository(repository.getOwnerNam
e(), repository.getName());
    ContentsService contentService = new
ContentsService(client);
    List<RepositoryContents> contentList =
contentService.getContents(repo);
    searchRecursive(contentList, repo,
contentService, dependencies);
}
```

Listing 1. *Metoda za pronalaženje zavisnosti*

U ovoj metodi se prvobitno dobavlja repozitorijum sa metodom *getRepository* na osnovu korisničkog imena vlasnika i naziva repozitorijuma, respektivno. Na kraju se poziva metoda *searchRecursive* koja rekurzivno prolazi kroz sve direktorijume unutar repozitorijuma kako bi pronašla fajl koji sadrži sve zavisnosti.

Ova metoda prima sledeće parametre: *contentList* – lista svih direktorijuma i fajlova unutar određenog direktorijuma repozitorijuma, repo – repozitorijum koji se skenira, *contentService* – servis koji omogućava dobavljanje svih direktorijuma i fajlova i *dependencies* – prazna lista u kojoj će se čuvati sve pronađene zavisnosti.

Kako je ovaj sistem ograničen samo na skeniranje *Maven* projekata, potrebno je vršiti rekurzivno kretanje sve dok se ne pronađe fajl pod nazivom *pom.xml* u kom su izlistane sve zavisnosti. Kada se fajl pronađe vrši se parsiranje i izvlačenje svih zavisnosti.

5.2 Komponenta za pretragu ranjivosti

Nakon što se pronađu sve zavisnosti koje se nalaze u repozitorijumu, vrši se pretraga ranjivosti. Pretraga ranjivosti vrši se nad NVD bazom podataka i to je implementirano uz oslonac na zvanični NVD API koji sadrži specifikaciju za način komunikacije sa bazom. Za pretragu nad bazom koristi se pretraga po ključnoj reči, što ovde predstavlja naziv zavisnosti. Za svaku zavisnost šalje se zahtev NVD bazi i dobija se odgovor u JSON formatu čija je šema definisana u specifikaciji NVD API-a. Ovaj JSON format dalje se mapira na model koji je definisan unutar sistema.

5.3 Komponenta za slanje izveštaja

Nakon uspešnog skeniranja repozitorijuma za zavisnosti i pretrage svih ranjivosti za njih, poziva se metoda koja generiše izveštaj koja se šalje korisniku na mejl. Generisani izveštaj sadrži sledeće informacije: naziv repozitorijuma, link ka repozitorijumu i listu svih pronađenih ranjivosti. Svaka ranjivost iz liste sadrži: naziv zavisnosti na koju se odnosi, CVE ID i link ka zvaničnom opisu ranjivosti unutar NVD baze.

6. ZAKLJUČAK

Tema ovog rada je kreiranje sistema za praćenje ranjivosti u softveru. Predstavljen je sistem koji na osnovu skeniranog repozitorijuma i izvlačenja svih zavisnosti kreira izveštaj sa listom javno poznatih ranjivosti vezane za pronađene zavisnosti.

Objašnjeni su neki od mehanizama za otkrivanje javno identifikovanih ranjivosti u softveru. Opisana je Lista ranjivosti (CVE), National Vulnerability Database (NVD), kao i efikasan sistem za ocenjivanje ranjivosti (CVSS).

Pored teorijskog dela, opisana su postojeća rešenja, model sistema i koraci u radu sistema. Prikazana je i implementacija sistema sa opisom tehnologija koje su korišćene za implementaciju, komponenta za skeniranje repozitorijuma na koju se ovaj sistem oslanja.

Preporuka za dalji razvoj veb-aplikacije je poboljšanje performansi pretrage javno identifikovanih ranjivosti. Pretraga se vrši putem NVD API-a gde je rezultat pretrage JSON datoteka sa svim ranjivostima koje su pronađene za zavisnost.

Nakon dobijenog rezultata parsira se JSON datoteka i mapira na model ranjivosti iz sistema. Ova faza oduzima najviše vremena u procesu analize. Razmotriti korišćenje *Elasticsearch*-a za skladištenje i pretragu celokupne NVD baze podataka.

Dalji razvoj veb-aplikacije može da ide u smeru uvođenja dodatnih vrsta projekata za skeniranje. Za sada je moguće samo skeniranje i izvlačenje zavisnosti iz Maven projekata.

7. LITERATURA

[1] Jose Carlos Coelho Martins da Fonseca, Marco Vieira, and Henrique Madeira, "Evaluation of Web Security Mechanisms Using Vulnerability & Attack Injection ", IEEE Transactions on depandable and soft computing, 2013.

[2] https://resources.whitesourcesoftware.com/blog-whitesource/owasp-a9-using-components-with-known-vulnerabilities (pristupljeno pristupljeno u maju 2021)

[3] Serena Elisa Ponta, Henrik Plate, "Detection assessment and mitigation of vulnerabilities in open source dependencies, 2020.

[4] Minzhe Gou, Ju An Wang, "An Ontology-based Approach to Model Common Vulnerabilities and Exposures in Information Security", ASEE Southeast Section Conference, 2009.

[5] https://ce.mitre.org/cve/identifiers/index.html, (pristupljen pristupljeno u maju 2021)

[6] Harold Booth, Doug Rike and Greg Witte, "The National Vulnerability Database (NVD): Overview", 2013.

[7] https://resources.whitesourcesoftware.com/blog-whitesource/the-national-vulnerability-database-explained, (pristupljeno u maju 2021)

[8] Peter Mell, Karen Scarfone, Sasha Romanosky, "An Analysis of CVSS Version 2 Vulnerability Scoring", Third International Symposium on Empirical Software Engineering and Measurement, 2006.

[9] Peter Mell, Karen Scarfone, Sasha Romanosky, A Complete Guide to the Common Vulnerability Scoring System Version 2.0, 2007.

[10] J. A. D. C. A. Jayakody, A. K. A. Perera, G. L. A. K. N. Perera, "Web-application Security Evaluation as a Service with Cloud Native Environment Support", International Conference on Advancements in Computing (ICAC), 2019.

[11] Dimitris Mitropoulos, Vassilios Karakoidas, Panos Louridas, Georgios Gousios, Diomidis Spinellis, "The Bug Catalog of the Maven Ecosystem", 2014.

[12] Ivan Pashchenko, Henrik Plate, Serena Elisa Ponta, Antonino Sabetta and Fabio Massacci, "Vulnerable Open Source Dependencies: Counting Those That Matter", Proceedings of the 12th International Symposium on Empirical Software Engineering and Measurement (ESEM), 2018.

[13]https://maven.apache.org/guides/introduction/introduction-to-the-pom.html (pristupljeno u junu 2021)

Kratka biografija:

Vladimir Cvetanović rođen je u Novom Kneževcu 29.09.1996. godine. Fakultet tehničkih nauka u Novom Sadu, smer Računarstvo i automatika, upisao je 2015. godine. Osnovne akademske studije završio je 2019. godine, nakon čega je upisao master akademske studije na Fakulte tehničkih nauka, smer Računarstvo i automatika.
kontakt: cvetanovic9696 @gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 007:004.056
DOI: https://doi.org/10.24867/15BE24Grujcic

EDITOR ZA XACML RBAC PROFIL

XACML RBAC PROFILE EDITOR

Nikola Grujčić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je analiziran XACML (OASIS eXtensible Access Control Markup Language) sa akcentom na RBAC (Role Based Access Control) profil ovog jezika. Objašnjeni su teorijski koncepti vezani za kontrolu pristupa. Razmatrano je definisanje prava pristupa kroz politike ovog profila, kojim bi se ispunili zahtevi za osnovni i hijerarhijski RBAC. Dat je opis implementirane aplikacije i prikaz reprezentativnih delova koda.*

Ključne reči: *XACML, RBAC, politika, pravilo, uloga, permisija, kontrola pristupa*

Abstract – *This paper analyzes XACML (OASIS eXtensible Access Control Markup Language) with an emphasis on the RBAC (Role Based Access Control) profile of this language. Terms and theoretical concepts related to access control are explained. The definition of access rights through policies was discussed, which would meet the requirements for basic and hierarchical RBAC. A description of the implemented application and a representation of representative parts of the code are given..*

Keywords: *XACML, RBAC, policy, rule, role, permission, access control*

1. UVOD

Kontrola i upravljanje pristupom informacijama je jedan od veoma bitnih problema svih informacionih sistema koji podržavaju veliki broj korisnika. Kako bi podaci ostali validni i kako bi se sprečila njihova zloupotreba, nameću se potrebe za zaštitom i obezbeđivanjem sigurnosti sistema. Jedno od mogućih načina da se ovaj problem reši jeste kontrola pristupa [1]. Ona predstavlja jedan od osnovnih koncepata bezbednosti čija je svrha ograničavanje operacija koje korisnici sistema mogu izvršiti nad podacima. Daljim razvojem kontrole pristupa, nastajali su različiti modeli koji bi definisali na koji način bi određeni korisnik mogao da pristupi određenom resursu sistema i na koji način bi mogao da ga koristi [1]. Jedan od modela kontrole pristupa koji se zasniva na konceptu uloga i privilegija naziva se *Role Based Access Control* (RBAC) [2]. Tema ovog rada je definisanje prava pristupa koristeći XACML *(extensible Access Control Markup Language)* jezik [3], oslanjajući se na RBAC profil ovog jezika. Ovaj rad opisuje veb editor namenjen za definisanje prava pristupa, adresira potencijalne probleme u pisanju prava pristupa na ovaj način i predlaže rešenja za unapređenje implementiranog softvera. u pisanju prava pristupa na ovaj način i predlaže rešenja za unapređenje implementiranog softvera.

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

2. KONTROLA PRISTUPA

Kontrola pristupa je jedan od osnovnih aspekata bezbednosti informacionih sistema [2]. Motivisana je potrebom da se pristup informacijama i dostupnim računarskim resursima i uslugama dozvoli samo ovlašćenim subjektima [4]. U poslednje tri decenije predloženi su različiti modeli kontrole pristupa [4]. Biće izdvojen poseban model kontrole pristupa koji se zasniva na ulogama, pošto je fokus rada usmeren ka ovom modelu. Nakon izlaganja ovih fundamentalnih stavki, analiziraće se krucijalni koncepti i ideje XACML-a.

2.1. Role Base Access Control

Kontrola pristupa zasnovana na ulogama (RBAC) je jedan od najpopularnijih modela kontrole pristupa za poslovne sisteme zbog svoje fleksibilnosti i skalabilnosti. Ovaj model se zasniva na individualnim ulogama određenog korisnika. Uloge su definisane u odnosu na radne odgovornosti osoba i bezbednosne politike organizacije. Kontrola pristupa zasnovana na grupama i ulogama može se uspešno primeniti u organizacijama sa precizno definisanom hijerarhijom vlasti i podelom dužnosti [2]. Centralna ideja RBAC modela jeste da su dozvole (eng. *Permissions*) povezane sa ulogama (eng. *Roles*), a korisnicima su pridružene odgovarajuće uloge. Stvaraju se uloge za različite funkcije posla u organizaciji i korisnicima se dodeljuju uloge na osnovu njihovih odgovornosti i kvalifikacija. Referentni RBAC model je definisan u okviru četiri modela [2, 4]:

1. osnovni RBAC,
2. hijerarhijski RBAC,
3. statičko razdvajanje dužnosti i
4. dinamičko razdvajanje dužnosti.

Osnovni (eng. *Core*) RBAC definiše minimalnu kolekciju RBAC elemenata, skupova elemenata i odnosa između elemenata kako bi se u potpunosti postigao sistem kontrole pristupa zasnovan na ulogama. To uključuje definisanje odnosa između korisnika i uloge, kao i odnosa između dozvola i uloga. Pored toga, uvodi koncept aktivacije uloga kao deo sesije korisnika u računarskom sistemu. Osnovni RBAC je neophodan u bilo kom RBAC sistemu, dok su ostali modeli nezavisni jedan od drugog. Hijerarhijska komponenta RBAC modela dodaje relacije za podržavanje hijerarhije uloga. Treća i četvrta komponenta modela dodaju relacije za postizanje ekskluzivnosti između uloga u skladu sa zadacima korisnika [2, 4].

2.2. Extensible Access Control Markup Language

XACML (eXtensible Access Control Markup Language) je jezički standard za izražavanje kontrole pristupa koji se

zasniva na XML (Extensible Markup Language) jeziku. Osmišljen je sa namerom da izrazi [5]:

- bezbednosne politike,
- zahteve za pristup koji su potrebni za postavljanje upita nad sistemom politika i
- odgovore sistema sa donošenom odlukom o ovlašćenju.

„Proširivi jezik za označavanje kontrole pristupa“ je jedan od načina izražavanja i mehanizama za sprovođenje kontrole pristupa koja se zasniva na atributima (eng. *Attribute-based access control* - ABAC). Razvijen je sa ciljem da se od operativnih okruženja aplikacije izdvoji proces kreiranja politika i proces donošenja odluka o ovlašćenju [6].

Većina operativnih okruženja implementira kontrolu pristupa na različite načine, svaki sa različitim opsegom kontrole i svaki u odnosu na različite tipove operacija i tipove izvora podataka. Ova heterogenost uvodi brojne administrativne izazove, pored samog izazova sprovođenja politike. Administratori su primorani da se izbore sa mnoštvom bezbednosnih domena pri upravljanju politikama pristupa i atributima. Budući da operativna okruženja implementiraju kontrolu pristupa na različite načine, teško je razmenjivati i deliti informacije o kontroli pristupa između različitih okruženja [6]. XACML nastoji da ublaži ove izazove stvaranjem zajedničkog i centralizovanog načina izražavanja svih podataka kontrole pristupa. Na osnovu zahteva za pristup koji dobija od aplikacija, XACML će donositi i sprovoditi odluke [5].

3. XACML RBAC PROFIL

XACML-ov profil za RBAC ispunjava uslove za osnovnu i hijerarhijsku kontrolu pristupa zasnovanu na ulogama [3]. U RBAC profilu XACML-a, telo za omogućavanje uloga (*Role Enablement Authority* - REA) je entitet koji korisnicima dodeljuje atribute i vrednosti, ili omogućava da u toku sesije korisnik poseduje odgovarajuće atribute i vrednosti kojima dokazuje da poseduje određenu ulogu. REA određuje koji skup uloga može biti dodeljene korisniku [3]. U nastavku će biti predstavljene vrste i svrha politika, koje mogu da se kreiraju. Biće objašnjen način kako da se postigne kontrola pristupa na osnovu ovog profila i adresiraće se ograničenja predloženog profila.

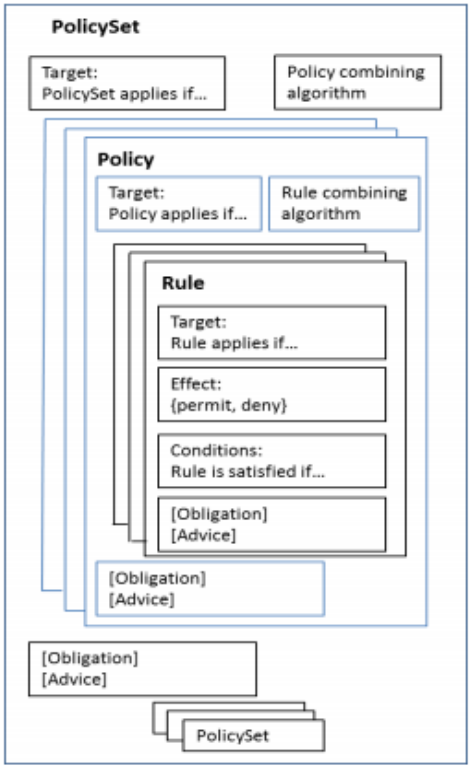
3.1. Politike

Zahtev za pristup se sastoji iz atributa subjekta (uglavnom korisnika koji je izdao zahtev), atributa resursa (resurs za koji se traži pristup), atributa vezanih za akciju (operacije koje treba izvesti na resursu) i atributa vezanih za trenutno okruženje [6]. XACML atributi se specificiraju svojim nazivom i vrednošću, pri čemu vrednosti atributa mogu biti različitih tipova. Svaki atribut označava svojstvo koje se odnosi na subjekat, odnosno resurs, akciju ili na okruženje [6]. Ukoliko uzmemo primer u bankarstvu, atributi uloge se mogu odnositi na vrstu pozicije (npr. blagajnik, službenik, šalterski radnik). Atributi resursa mogu se odnositi na podatke o računima, kreditima itd. Pod okruženjem se može navesti vreme prijema stranke, broj šaltera itd.

Atributi subjekta i resursa se skladište u repozitorijumima. Za razliku od atributa subjekata i resursa koji se moraju uneti administrativnim putem, atributi okruženja zavise od dostupnosti sistemskih senzora i mogu se odnositi na trenutno vreme, dan u nedelji, a može predstavljati i nivo pretnje na sistem.

Na slici 1 ilustrovan je sadržaj XACML politike. Politike pristupa su strukturirane kao skupovi politika. U ovom skupu se mogu naći pojedinačne politike ili drugi skupovi politika. Pravilo je elementarna jedinica politike i ne može da egzistira nezavisno od politike [5]. Budući da nisu sva pravila, politike ili skupovi politika relevantni za svaki zahtev, XACML uključuje element *Target*. Ovaj element definiše jednostavan logički uslov kojim se utvrđuje primenljivost datog skupa politika, odnosno pojedinačne politike ili pravila na konkretni zahtev korisnika [5]. Radi jednostavnosti i razumljivosti, u radu će se pod pojmom politika, pored pojedinačne politike, podrazumevati i skup politika.

Pored *Target* elementa, pravilo uključuje i niz logičkih uslova na osnovu kojih se određuje efekat pravila, tj. odobrava ili odbija zahtev. Svi logički uslovi moraju biti zadovoljeni kako bi se zahtev odobrio. Logički uslovi pravila mogu da budu kompleksniji od uslova u *Target* elementima, jer mogu uključivati složenije funkcije, poput funkcija poređenja vrednosti i kombinacija drugih funkcija. Navode se u *Condition* elementu [5].



Slika 1. *XACML politika* [6]

3.2. Vrste politika

Postoje četiri vrste politika koje su specificirane u ovom profilu. To su [3, 7]:

- *Role <PolicySet>* - politika koja povezuje vlasnike datog atributa i vrednosti uloge sa permisijama za tu ulogu. Svaka ova politika upućuje na jednu odgovarajuću politiku sa

skupom permisija, ali ne sadrži niti upućuje na druge tipove politika.

- *Permission* <*PolicySet*> - politika koja sadrži dozvole povezane sa datom ulogom. Sadrži druge politike kojima se opisuju resursi kojima subjekti imaju dozvoljen pristup. Takođe, precizira i sve uslove tog pristupa i akcije koje se dozvoljene. Data politika može da sadrži reference na druge politike ovog tipa koje su povezane sa drugim ulogama. Na ovaj način se dozvoljava da politika nasledi sve dozvole povezane sa drugom ulogom.
- *HasPrivilegesOfRole* <*Policy*> - politika koja može biti sadržana u *Permission* politici. Ova politika podržava zahteve koji pitaju da li subjekat ima privilegije date uloge.
- *Role Assignment* <*Policy*> ili <*PolicySet*> - politika koja definiše koje se uloge mogu omogućiti ili dodeliti određenom subjektu. Ovom politikom se mogu specificirati ograničenja na različite kombinacije uloga ili ukupan broj uloga koje subjekat može da poseduje.

Politike navedene u ovom profilu mogu odgovoriti na dve vrste pitanja [3]:

- Ako subjekat ima omogućene uloge R1, R2, ... Rn, može li subjekat izvršiti datu akciju nad datim resursom?
- Ako subjekat ima omogućene uloge R1, R2, ... Rn, znači li to da će subjekat imati dozvole povezane sa datom ulogom R'? Odnosno, da li je uloga R' jednaka ili junior od bilo koje od uloga R1, R2, ... Rn?

Dolazimo do zaključka da politike ovog profila ne odgovaraju na pitanje: „Koji skup uloga ima subjekat?“ To pitanje mora rešavati telo za omogućavanje uloga. Pretpostavlja se da su subjektu sve uloge već omogućene u vreme kada se traži odluka o autorizaciji. Takođe, ne bave se okruženjem u kojem se uloge moraju dinamički omogućiti na osnovu resursa ili radnji koje subjekat pokušava da izvrši. Zaključujemo da politike navedene u ovom profilu se ne bave statičkim ili dinamičkim razdvajanjem dužnosti, koji je predviđen RBAC modelom [3].

3.3. Kontrola pristupa

Kontrola pristupa zasnovana na ulogama može da se implementira oslanjajući se na dva tipa politika [3, 7]:

- *Role* politike i
- *Permission* politike.

Potrebno je da za svaku ulogu bude definisana jedna *Role* politika koji sadrži *Target* element koji čini ovu politiku primenljivom samo na subjekte koji imaju XACML atribut pridružen datoj ulozi. Element *Target* ne sme ni na koji način da ograniči resurs, akciju nad resursom ili okruženje [3]. Svaka *Role* politika treba da sadrži jedan *PolicySetIdReference* element koji upućuje na *Permission* politiku sa dozvolama date uloge. Pored toga, *Role* politika ne sme da sadrži nijednu drugu politiku ili da referencira druge politike [3]. Za svaku ulogu treba da se definiše jedna *Permission* politika. Ova politika treba da sadrži *Rule* elemente kojim se specificiraju tipovi pristupa dozvoljeni subjektima koji imaju datu ulogu [3, 7].

4. SPECIFIKACIJA SISTEMA

Softversko rešenje predstavlja veb editor u kojem se definišu prava pristupa oslanjajući se na RBAC profil XACML-a. Osnovni zadatak ovog sistema je da obezbedi proces kreiranja politika kojima bi se zadovoljili zahtevi za definisanje osnovnog i hijerarhijskog RBAC-a. Rešenje se bazira na specifikaciji standarda XACML verzije 3.0 (XACML v3.0 Core and Hierarchical Role Based Access Control (RBAC) Profile Version 1.0) [3].

4.1. Arhitektura sistema

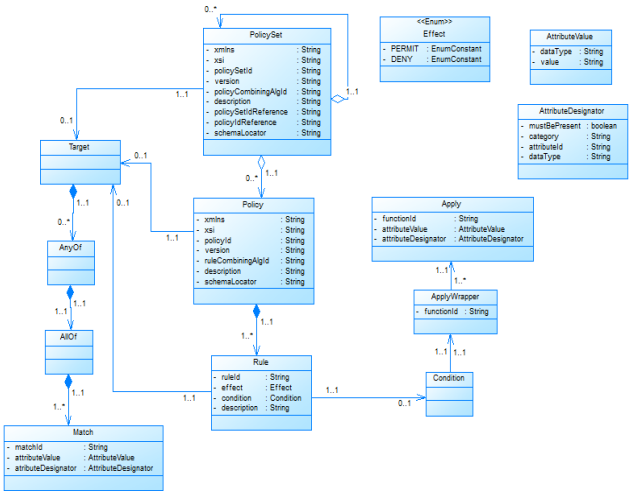
Sistem za kreiranje i editovanje XACML politika se sastoji iz dva modula:

1. korisnički modul i
2. *back-end* aplikaciju.

Korisnički modul predstavlja *front-end* aplikaciju koja obezbeđuje interfejs za krajnjeg korisnika i omogućuje mu sve neophodne funkcionalnosti. Predstavlja *single-page* aplikaciju koja je realizovana upotrebom *Angular* okvira 8.0.6. Poslovna logika sistema je smeštena u *back-end* modulu. Realizovan je kao servisno orijentisana veb aplikacija koja je implementirana u *Java Spring* okviru. Komunikacija između klijentskog i serverskog modula se obavlja putem REST tehnologija. Sav saobraćaj vršen je posredstvom HTTP protokola. *Hibernate* biblioteka se koristi za manipulaciju podacima koji su smešteni u *MySQL* bazi. Pored relacione baze podataka, korišćena je i *MongoDB* baza podataka, kako bi bili ispunjeni zahtevi čuvanja politika u vidu dokumenata. Na *back-end* aplikaciji se koristi *Spring Security* okvir kojim se obavlja autentifikacija korisnika.

4.2. Model podataka

Kako bi se stekao bolji uvid u način organizacije i funkcionisanja sistema, na slici 2 je kroz dijagram klasa prikazan odnos između relevantnih entiteta koji su nam potrebni kako bismo kreirali željene politike.



Slika 2. *Isečak iz dijagrama klasa*

Tek nakon parsiranja u XML format, politike će biti sačuvane u nerelacionoj bazi podataka. *PolicySet* entitet će se sastojati iz instanci *Policy* entiteta, a pored toga može obuhvatati i druge instance *PolicySet* entiteta. *Policy* se sastoji iz *Rule* entiteta. Instance *Policy* entiteta će moći da nastave da egzistiraju ukoliko *PolicySet* više ne bude postojao.

Nasuprot tome, instance *Rule* klase neće moći da egzistiraju ukoliko se obriše roditeljska instanca *Policy* entiteta. Svaki od prethodno navedenih entiteta će sadržati *Target* element. Ovaj element može biti prazan ili može sadržati *AnyOf* instance. Dalje, *AnyOf* instanca mora da sadrži *AllOf* instancu, koja je neophodno da sadrži *Match* instancu. *Rule* entitet sadrži *Condition*. Ovaj element će biti sačinjen iz *Apply* elemenata.

5. IMPLEMENTACIJA SISTEMA

U ovom poglavlju će biti opisane funkcionalnosti čija implementacija je bila neophodna, kako bi se omogućio proces kreiranja i editovanja željene politike.

Nakon uspešne registracije i prijave na sistem, korisnik će imati mogućnost kreiranja politika. Ova proces će se sastojati iz nekoliko *Reactive* formi, kako bi se uneli svi potrebni podaci i kako bi interakcija korisnika sa sistemom bila prirodna i jasna.

Kako bi podaci, prilikom kreiranja politika, bili dostupni unutar aplikacije sa centralizovanog mesta, na klijentskoj strani je potrebno koristiti odgovarajući sistem za upravljanje stanjem aplikacije. Ovaj problem je rešen pomoću *Store* biblioteke, koju obezbeđuje *NgRx* (*Angular Reactive Extensions*). Na ovaj način je omogućena obrada i skladištenje podataka na centralizovanom mestu unutar aplikacije i praćenje promena prouzrokovanih događajima. Radi lakšeg snalaženja i bolje preglednosti podataka, a i pošto je XML moguće vizualizovati kroz strukturu stabla, autor se opredelio da koristi *TreeGrid* komponentu iz *Syncfusion Angular UI* biblioteke komponenata. Ovom komponentom je omogućeno dinamičko dodavanje elemenata u stablo, proširivanje i skupljanje zapisa prikazanih u stablu, što odgovara potrebama projekta.

Kako bismo zapisali politiku u XML formatu, potrebno je da se koristi odgovarajući parser. Izabran je JAXB (*Java Architecture for XML Binding*) parser.

U nastavku će biti prikazana politika koja je kreirana implementiranim rešenjem. Pretpostavimo da bi u organizaciji postojala uloga *employee* koja bi imala dozvolu za čitanje politika. Neophodno je definisati jednu *Role* i *Permission* politiku. *Permission* politika koja sadrži permisiju vezanu za datu ulogu je prikazana na slici 3.

```
<PolicySet PolicyCombiningAlgId="&policy-combine:permit-overrides"
    Version="0.1"
    PolicySetId="PPS:employee:role"
    xsi:schemaLocator="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17
        xacml-core-v3-schema-wd-17.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17">
    <Policy Version="1.0" xsi:schemaLocator="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17
        xacml-core-v3-schema-wd-17.xsd"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
        RuleCombiningAlgId="&rule-combine:permit-overrides"
        PolicyId="Permissions:of:employee:role">
        <Rule Effect="Permit" RuleId="Permission:to:read:policy">
            <Target>
                <AnyOf>
                    <AllOf>
                        <Match MatchId="&function:string-equal">
                            <AttributeValue
                                DataType="&xml:string">read policy</AttributeValue>
                            <AtributeDesignator
                                DataType="&xml:string"
                                AttributeId="&action:action-id"
                                Category="&category:action"
                                MustBePresent="false"/>
                        </Match>
                    </AllOf>
                </AnyOf>
            </Target>
        </Rule>
        <Target/>
    </Policy>
    <Target/>
</PolicySet>
```

Slika 3. *Primer Permission politike*

Ova permisija će dozvoliti bilo kojem subjektu da obavi akciju čitanja politike, ukoliko se dodeli odgovarajućoj *Role* politici.

6. ZAKLJUČAK

U radu je predloženo softversko rešenje za definisanje prava pristupa putem politika, oslanjajući se na RBAC profil XACML jezika. Na osnovu ovih politika i uz pomoć odgovarajućeg mehanizma za sprovođenje kontrole, moguće je obezbediti kontrolu pristupa u informacionim sistemima. Obrađene su osnovne ideje na kojim se bazira kontrola pristupa, analiziran je XACML jezik i njegov mehanizam sprovođenja kontrole pristupa. Ove teorijske koncepte je bilo neophodno izložiti pre samog analiziranja RBAC profila XACML-a. Takođe, opisano je implementirano softversko rešenje i prikazana je njegova arhitektura.

Trenutno implementiranim rešenjem je ograničen pristup i editovanje politike na kreatora politike. Jedno od mogućih proširenja ovog rešenja jeste podrška da više korisnika jedne organizacije mogu da kreiraju i preuređuju politike. Ukoliko bi se ovo proširenje podržalo, bilo bi neophodno uvesti i podršku za verzioniranje politika. Verzioniranjem bi se obezbedilo da stanje politike uvek ostane konzistentno i osigurala bi se potpuna kontrola nad procesom kreiranja politika. Takođe, korisniku bi bila dostupna i istorija svih izmena nad politikama.

7. LITERATURA

[1] A. Poniszewska-Maranda, Management of access control in information system based on role concept, 2011.

[2] D. R. K. R. C. David F. Ferraiolo, Role-Based Access Control - Second edition, 2007.

[3] H. Bill Parducci, XACML v3.0 Core and Hierarchical Role Based Access Control RBAC) Profile Version 1.0 Committee Specification 02, 2014.

[4] S. I. T. I. Council, American National Standard for Information Technology – Role Based Access Control, 2004.

[5] The eXtensible Access Control Markup Language (XACML), Version 3.0, OASIS Standard, 2013.

[6] R. C. R. K. a. V. H. David Ferraiolo, Extensible Access Control Markup Language (XACML) and Next Generation Access Control (NGAC), 2016.

[7] S. M. Anne Anderson, Core and Hierarchical Role Based Access Control (RBAC) profile of XACML v2.0, 2005.

Kratka biografija:

Nikola Grujčić rođen je je u Zrenjaninu 1996. godine. Osnovne akademske studije završio je 2019. godine na Fakultetu tehničkih nauka. Master rad iz oblasti Elektrotehnike i računarstva – Računarstvo i automatika je odbranio 2021. godine

kontakt: nikolagrujcic@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.38
DOI: https://doi.org/10.24867/15BE25Omcikus

REALIZACIJA INDUSTRIJSKOG POGONA PRIMENOM STEP MOTORA

IMPLEMENTATION OF INDUSTRIAL DRIVE USING STEP MOTORS

Vladimir Omčikus, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Rad se bavi realizacijom upravljanja step motorima na primeru pogona za serijalizaciju (obeležavanje) kutija lekova. U teorijskom delu rada obrađeni su uopšteno step motori, način rada, podela i načini upravljanja. Pored toga obrađena je hardverska realizacija pogona sa akcentom na drajv koji se koristi u samom pogonu, njegovim načinima upravljanja i modovima rada. Praktični deo rada se bavi realizacijom same profibus komunikacije i realizacijom softverskog dela za upravljanje step pogonom.*

Ključne reči: *Step motori, Profibus, Upravljanje*

Abstract – *The topic of the paper is the realization of step motors control on the example of a plant for serialization (marking) of drug boxes. In the theoretical part of the paper, step motors, mode of operation, division and control methods are discussed in general. In addition, the hardware realization of the drive is processed, with an emphasis on the drive used in the drive and its control modes and operating modes. The practical part of the paper deals with the realization of profibus communication and the softvers realization for step drive management.*

Keywords: *Step motors, Profibus, Management*

1. UVOD

Elektromotorni pogon (EMP) je elektromehanički sistem koji električnu energiju iz primarnog izvora na kontrolisan način pretvara u mehanički rad, i time služi za pokretanje opreme ili opterećenja.

Razlikujemo dve vrste elektromotornog pogona, to su: neregulisani i regulisani pogoni. Neregulisani EMP su pogoni kod kojih nije potrebno menjanje brzine ili stalno pokretanje i zaustavljanje kao što su pumpe, ventilatori i mlinovi. Regulisani elektromotorni pogon (REMP) je EMP u kome je moguće kontrolisati ostvaren moment, brzinu ili položaj motora. Step motori se uglavnom koriste kao regulisani elektromotorni pogoni. Kod regulisanog EMP-a sa step motorima razlikujemo dve vrste pogona: bez povratne sprege i sa povratnom spregom.

Tema rada jeste realizacija upravljanja step motorima na primeru pogona za obeležavanje i serijalizaciju kutija za lekove u farmaceutskoj industriji.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Zoltan Čorba, vanr. prof.

Nakon uvodnog dela u drugom poglavlju obrađeni su uopšteno step motori, način rada, podela i načini upravljanja. U trećem poglavlju obrađena je hardverska realizacija samog pogona sa akcentom na drajv koji se koristi u pogonu, njegovim načinima upravljanja, modovima rada. Četvrto poglavlje se bavi realizacijom same profibus komunikacije i praktičnom realizacijom softverskog dela za upravljanje step pogonom.

2. STEP MOTORI

Step motor je "digitalna" verzija elektromotora. On je elektromehanički uređaj koji pretvara električne impulse u mehanički pomeraj, najčešće su rotacioni a ređe translatorni.

Široko prihvaćen step motor u poslednje dve decenije podstaknut je usponom digitalne elektronike. Savremena elektronika i primena drajvera sa mikrokontolerima lako se prilagođavaju zahtevima koji su potrebni za rad ovakvih uređaja. Step motori su robusni i jeftini jer rotor ne sadrži namotane klizne prstenove ili komutator. Rotor je cilindrične čvrste građe, koji takođe može imati istaknute polove ili fine zube, vrlo je često permanentni magnet.

Pogodni su za aplikacije gde su potrebna kompaktna i robusna rešenja. Oni razvijaju maksimalni obrtni moment pri mirovanju što ih čini prirodno pogodnim za držanje položaja. Spoljna komutacija osigurava da je brzina savršeno konstantna čak i ako opterećenje varira.

Zahvaljujući odsustvu bilo kojih elektronskih komponenti, step motori rade tamo gde senzori ili merači položaja drugih vrsta motora pokazuju svoje nedostatke pri teškim uslovima rada, kao što su: visoke/niske temperature, vibracije, buka i prašina.

2.1. Vrste step motora

Zajednička karakteristika svih step motora je sekvencijalno pobuđivanje njihovih statorskih namotaja električnim impulsima, pri čemu korak motora zavisi od broja faza i polova statora, odnosno magnetnih osobina i broja polova rotora.

Prema izvedbi statorskog namotaja i načinu upravljanja, razlikujemo dve vrste:

- Unipolarni step motori,
- Bipolarni step motori.

Step motori se po načinu gradnje dele u tri osnovne kategorije:

- Step motori sa stalnim (permanentnim) magnetima,

- Hibridni step motori,
- Step motori sa promenljivom reluktansom.

3. REALIZACIJA STEP POGONA NA PRIMERU SISTEMA ZA SERIJALIZACIJU I OBELEŽAVANJE

Praktična tema rada je sistem za serijalizaciju i obeležavanje kutija za lekove u farmaceutskoj industriji, tačnije samo deo koji se odnosi na step pogone , slika 1.

Glavne uređaje sistema čine: dve etiketirke koje služe za lepljenje sigurnosnih markica, štampač za štampanje jedinstvenih kodova, kamere za čitanje i proveru koda. Sistem je potpuno automatizovan, dovoljno je da operater na operatorskom panelu izabere kutiju, i na osnovu nje će se ceo sklop automatski pozicionirati. Ceo sistem poseduje dvanaest step motora, koji su umreženi profibus komunikacijom sa PLC-om. Kad operater izabere prozivod, gornji konvejer, konvejeri sa strane, pogon kamere, pogon etiketirke i pogon glave štampača će se postaviti u poziciju koja je optimalna za izabranu kutiju.

Slika 1. *Sistem za serijalizaciju i obeležavanje u farmaceutskoj industriji*

Putanja kutija je sledeća: Prvo se vrši obeležavanje, odnosno ispisivanje koda pomoću štampača, zatim kutija prolazi ispred kamere koja čita kod i smešta ga u bazu. Ukoliko je kod nečitljiv ili nije dobar, PLC šalje signal za odbacivanje ka duvaljci, koja se nalazi na samom izlazu. Posle kamere kutija prolazi pored dve etiketirke koje se nalaze sa obe strane kutije i vrše apliciranje sigurnosnih markica. Samo u slučaju da je kod verifikovan od strane kamere vrši se apliciranje sigurnosnim markicama. Nakon apliciranja vrši se provera pozicije i kvaliteta zalepljenosti markica pomoću dva optička senzora sa optičkim vlaknima. Kada markica nije aplicirana ili nije dobro zalepljena, kutija će biti odbačena. Ukoliko je kutija prošla sve provere ona dolazi do izlaznog konvejera.

3.1. Drajv za step pogon

Step pogon sa slike 1, se sastoji od step drajva SD328B i trofaznog step motora BRS3. Referentne vrednosti se zadaju i kontrolišu preko PLC-a ili namenskog kontrolera. Ovaj step pogon ima odlične brzinske karakteristike. Zbog velikog obrtnog momenta pri malim brzinama rotacije je posebno pogodan za pozicioniranje na kratkim udaljenostima. Još jedna prednost je visok obrtni momenat u mirovanju.

Drajveri koji se nalaze u ovom pogonu imaju mogućnost profibus komunikacije i zajedno sa PLC-om se vrši kontrola i upravljanje ovim pogonom.

3.2 Operativni modovi rada drajva

U SD328B step drajveru su integrisani različiti operativni modovi koju mu omogućavaju široku primenu u industriji, kao što su:

- Ručni mod (JOG) za poziciju ili brzinu,
- Mod elektronskog reduktora,
- Kontrola pozicije,
- Kontrola brzine,
- Homing.

Ručni mod (JOG) - Ovaj mod omogućava da se po jednoj osi vrši pomeranje ručno. Kretanje se može ostvariti kao jedan pomeraj (pozicioni JOG) i kontinualno pri konstantnoj brzini (brzinski JOG). Na raspolaganju su dve brzine, sporo ili brzo.

Mod elektronskog reduktora - U režimu rada elektronskog reduktora referentni signal dolazi u formi A/B signala enkodera ili signala impuls/smer P/D. Nova pozicija se računa na bazi impulsnih signala i promenljivog prenosnog odnosa.

Kontrola pozicije - Ovaj mod sa podesivim kretanjem se izvodi od početne pa do krajnje pozicije. U slučaju apsolutnog pozicioniranja, pozicija je definisana apsolutno u odnosu na nultu tačku ose. Nulta tačka se mora definisati prilikom definisanja hominga, pre prvobitne upotrebe apsolutnog pozicioniranja. U slučaju relativnog pozicioniranja pozicija se određuje relativno u odnosu na trenutnu poziciju ili na ciljanu poziciju.

Kontrola brzine - U ovom režimu rada podešava se referenca za brzinu motora i započinje se kretanje bez ciljane pozicije. Ova brzina se održava sve dok se ne promeni zadata referenca ili se ne pređe u drugi operacioni mod.

Homing - Homing operacija se sastoji od pridruživanja pozicije ose sa poznatom mehaničkom pozicijom. Ova pozicija postaje referentna pozicija za svako pomeranje po osi. Postoje dve vrste hominga koji se zasnivaju na: referentnom kretanju i pozicionom podešavanju.

Referentno kretanje je kretanje ka referentnoj tački na osi, cilj je utvrđivanje apsolutnog referentnog položaja između položaja motora i položaja ose. Referentna tačka takođe definiše nultu tačku koja se koristi kao referentna tačka za sva naknadna pomeranja. Svako referentno pomeranje mora da dosegne referentu tačku da bi bilo validno, ako je referentno kretanje iz bilo kog razloga prekinuto, mora se započeti opet.

Poziciono podešavanje se može koristiti za kontinuirano kretanje motora bez prekoračenja ograničenja pozicioniranja. Homing poziciono podešavanje se može izvršiti samo kada je motor u stanju mirovanja. Ovaj postupak sprečava prekoračenje apsolutnih ograničenja položaja tokom pozicioniranja, jer se nulta tačka postavlja neprekidno.

4. REALIZACIJA UPRAVLJANJA STEP POGONOM NA PRIMERU SISTEMA ZA SERIJALIZACIJU I OBELEŽAVANJE

Upravljanje step drajverima SD328 se vrši pomoću fieldbus mreže. U ovom pogonu koristimo seriju SD328B, sa kojom se upravlja pomoću PROFIBUS-a.

4.1. Profibus tehnologija

Profibus je pametna komunikaciona tehnologija. Uređaji u sistemu su povezani sa centralnom linijom. Jednom povezani, ovi uređaji mogu da razmenjuju informacije na efikasan način. Uređaji koji koriste ovaj način komunikacije mogu učestvovati u samodijagnostici i dijagnostici povezivanja. Profibus koristi topologiju magistrale (eng. bus topolgy). Kod ove topologije, centralni vod ili magistrala, prolazi kroz čitav sistem, dok su svi uređaji redno povezani sa njim. Ovakav sistem eliminiše potrebu da vodovi pune dužine idu od kontrolera do svakog pojedinačnog uređaja.

Vrste Profibus-a:

- Profibus FMS - Početna verzija Profibus-a bio je Profibus FMS (eng. Fieldbus Message Specification). Profibus FMS je dizajniran za komunikaciju između programibilnih kontrolera i računara. Nažalost, kao početni napor Profibus dizajnera, FMS tehnologija nije bila dovoljno fleksibilna. Ovaj protokol nije bio pogodan za manje kompleksnu komunikaciju na širim komplikovanijim mrežama. Novi tipovi Profibus-a će zadovoljiti ove potrebe.
- Profibus PA - Profibus PA (eng. Process Automation) je protokol osmišljen za procesnu automatizaciju. Profibus PA standardizuje proces prenosa mernih podataka i posebno je kreiran za upotrebu u opasnim okruženjima.
- Profibus DP - Profibus DP (eng. Decentralized Periphery) je protokol koji je posebno prilagođen za komunikaciju u proizvodnim procesima. Njegove osnovne karakteristike su brzo i jednostavno umrežavanje dodatnih uređaja i velika brzina prenosa. Profibus DP postoji u tri različite verzije: DP-V0, DP-V1 i DP-V2. Profibus DP verzija DP-V0 je protokol koji je korišten za komunikaciju sa step driverima u ovom industrijskom pogonu.

4.2. Realizacija Profibus DP topologije

Za realizaciju upravljanja pogonom, koji je tema ovoga rada, koristi se Profibus DP-V0 tehnologija. Profibus magistrala se sastoji od PLC-a kao DP master uređaja i dvanaest drajvera kao podređenih uređaja. U ovom pogonu master uređaj je Siemens PLC CPU 1214C, DC / DC / DC, 14DI / 10DO / 2AI , dok se za komunikaciju koristi Profibus DP kartica Siemens CM 1243-5.

Umrežavanje uređaja se vrši sa jednim Profibus kablom, koji ide iz komunikacione kartice ka drajverima. Svaki drajver ima ulazne i izlazne terminale, što omogućava vezivanje svih podređenih uređaja redno, jedan za drugim, slika 2.

Slika 2. *Topologija Profibus DP*

Master uređaj koristi identifikacioni broj za identifikaciju podređenog uređaja. Idenifikacioni broj koji je vezan za jedan uređaj je jedinstveni broj dodeljen automatski od strane master uređaja, to u suštini predstavlja adresu uređaja. Svakom uređaju se dodeljuje jedinstvena adresa 1...126, podređeni uređaju najčešće koriste adrese 3...126, dok su za master uređaje rezervisane adrese 0...2. Po potrebi adrese se mogu menjati i ručno.

Profibus DP V0 obezbeđuje bazičnu funkcionalnost DP-a. To uključuje cikličnu razmenu podataka, kao i dijagnostičke i procesne alarme uređaja konektovanih na magistrali. Master uređaj ciklično upisuje i šalje podatke ka podređenim uređajima, na isti način se vrši i čitanje pristiglih podataka sa podređenih uređaja.

Razmena podataka sledi fiksni obrazac:

- Slanje podataka ka podređenom uređaju: Master uređaj smešta podatak u memoriju namenjenu za slanje podataka. Sa te memorijske lokacije se šalje ka podređenom uređaju i izvršava.
- Primanje podataka od podređenog uređaja: Master uređaj zaprima podatak o izvršenju komande od strane podređenog uređaja. Ukoliko master uređaj zaprimi potvrdnu poruku bez ikakve dodatne poruke o grešci, to znači da je zadata komanda izvršena. Master uređaj može da šalje novu komandu.

4.3. Realizacija upravljanja primenom korisničkog interfejsa

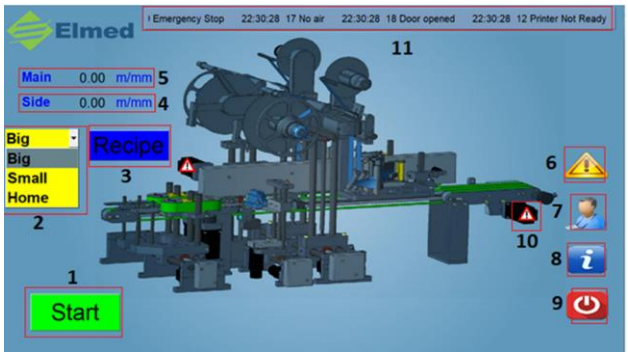
Kao što je već napomenuto u prethodnom delu, sistem ima dvanaest step motora. Oni se pokreću sa upravljačkog panela pritiskom na dugme “START” i zaustavljaju pritiskom na dugme “STOP”. Od dvanaest motora pet se koriste za pokretanje konvejera, dok se sedam motora koristi za pozicioniranje.

Svaki motor koji radi u pozicionom modu ima induktivni homing senzor koji predstavlja referencu na osnovu koje se motor pomera na željenu poziciju. Prilikom svake promene proizvoda sistem će automatski ići u homing i motori će se pozicionirati tako da to odgovara trenutnom proizvodu. Homing mod se aktivira i u slučaju greške u sigurnosnom krugu a nakon reseta, kao i u slučaju nepredviđenog isključenja sistema ili nestanka struje,

nakon toga motori će se pozicionirati po receptu za poslednji izabrani proizvod.

Prilikom pokretanja sistema potrebno je odabrati ili proveriti tip proizvoda koji se trenutno radi, svaki put kada se odabere različit proizvod od prethodnog sitem će automatski krenuti u homing i novo pozicioniranje. Takođe postoje opcije ponovnog repozicioniranja za izabrani proizvod ili opcija da se motori postavite u samu homing poziciju, slika 3. Ukoliko se motori žele postaviti u homing poziciju to se vrši izborom opcije "Home" iz padajućeg menija. Dok je za ponovno repozicioniranje potrebno je pritisnuti taster "Recepie".

Ukoliko je izabran proizvod i ciklus pozicioniranja završen unosi se vrednost brzine transportera, nakon toga je moguće pokrenuti sistem pritiskom na taster "Start". Zaustavljanje sistema se vrši na istom mestu pritiskom na isti taster, samo će u ovom slučaju biti označen sa "Stop", slika 3.



Slika 3. *Glavni ekran upravljačkog panela*

Na glavnom ekranu se nalazi:

1. START/STOP mašine,
2. Izbor proizvoda/postavljanje motora u home poziciju,
3. Komanda za pozicioniranje
4. Upisuje se vrednost brzine glavnog i transportera iznad,
5. Upisuje se vrednost brzine transportera za vođenje sa strane,
6. Lista alarma,
7. Logovanje korisnika,
8. Informacije (naziv sistema, informacije o firmi),
9. Gašenje računara,
10. Signalizacija greške pogona,
11. Prikaz trenutnih alarma.

5. ZAKLJUČAK

Koračni motori su danas jako zastupljeni u različitim granama proizvodnje, uprkos svojim manama ovi motori svaki dan se sve više usavršavaju i šire oblast svoje primene. Njihova upotreba pretežno zavisi od očekivanja i primene. Ako se teži visokim preciznostima, mehaničkoj čvrstoći, prilagođenosti korisnicima, kontroli i održavanju položaja rotora bez napajanja, onda je ova tipologija motora idealna.

Step pogoni su odlični za aplikacije pozicioniranja. Step motori se mogu precizno kontrolisati u smislu udaljenosti i brzine jednostavnim variranjem broja impulsa i njihove frekvencije. Njihov veliki broj polova daje im tačnost, dok istovremeno mogu da rade u otvorenoj petlji. Ako je odgovarajuće snage i momenta za izabranu aplikaciju, step motor nikada neće propustiti korak. Pošto im nije potrebna poziciona povratna informacija, a samim tim i dodatna oprema za pozicioniranje, vrlo su isplativi.

Međutim, trebalo bi uzeti u obzir neka ograničenja: nemogućnost rada na velikim brzinama, mogućnost gubljenja koraka a da se to ne primeti na vreme (npr. u slučaju nekog mehaničkog problema), prilično se zagrevaju u radu (radna temperatura oko 90°C).

Izbor step pogona kao EMP-a sistema za serijalizaciju i obeležavanje pokazao se kao jako dobro rešenje. Motori koji se koriste za pozicioniranje su odgovarajući, prevashodno u pogledu preciznosti i snage. Svaki od tih motora pokreće vreteno preko kaišnog prenosa pa se na taj način dobija dodatna snaga i dodatan moment. Motori koji se koriste za pokretanje konvejera pokazali su se takođe kao odlično rešenje, i na većim brzinama do 70 m/min.

Pogon se pokazao kao prilično kompaktan i jednostavan za upravljanje i rad sa strane operatera. Svi motori, kaišni prenosnici, senzori i ostala oprema su pristupačni u slučaju potrebnog servisiranja i održavanja.

6. LITERATURA

[1] Schneider Electric "SD328B Stepper motor drive - Product manual, V2.03, 07.2010".

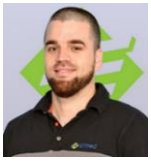
[2] Schneider Electric "SD328A CANopen DSP402, Fieldbus interface - Fieldbus manual, V2.01, 11.2008".

[3] A. Sugawara, T. Kenjo, "Stepping Motors and Their Microprocessor Controls", Oxford University Press.

[4] Stojanovič D., "Koračni motori – studija konstruktivnih i funkcionalnih karakteristika", Tehnički fakultet, Čačak, 1995.

[5] https://www.automate.org/industry-insights/bringing-closed-loop-functionality-to-stepper-motors

Kratka biografija:



Vladimir Omčikus rođen je u Šibeniku 02.08.1987. Srednju elektrotehničku školu "Mihajlo Pupin", završio je u Novom Sadu 2006 god. Fakultet tehničkih nauka, master studije, studijski program Energetika, elektronika i telekomunikacije upisao je školske 2015/16. Na studijama se opredelio za smer Elektroenergetika – Energetska elektronika i električne mašine.

Mail: vladimiromcikus@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.39
DOI: https://doi.org/10.24867/15BE26Gojic

ARHITEKTURA I BEZBJEDNOSNI ZAHTJEVI 5G MOBILNIH MREŽA

ARCHITECTURE AND SECURITY REQUIREMENTS OF 5G MOBILE NETWORKS

Jovan Gojić, Željen Trpovski, Dejan Nemec, *Fakultet tehničkih nauka, Novi Sad*

Oblast – TELEKOMUNIKACIONI SISTEMI

Kratak sadržaj – *U radu su opisane dvije ključne tehnologije koje se koriste u arhitekturi 5G sistema, a to su: softversko definisano umrežavanje (SDN) i virtualizacija mrežnih funkcija (NFV). Pored toga, prikazani su konkretni primjeri ugrožavanja bezbjednosti usluge, kao i bezbjednosni mehanizmi koji će omogućiti da korisnik pristupa uslugama bez rizika od gubitka autentičnosti, povjerljivosti, dostupnosti usluge i integriteta.*

Ključne riječi: *softversko definisano umrežavanje (SDN), virtualizacija mrežnih funkcija (NFV), 5G mobilna mreža*

Abstract – *The paper presents two key technologies used in the architecture of fifth generation networks, namely: software defined networking (SDN) and virtualization of network functions (NFV). In addition, concrete examples of endangering the security of the service are presented, as well as security mechanisms that will allow the user to access the services without the risk of losing the authenticity, confidentiality, availability of the service and integrity.*

Keywords: *software defined networking (SDN), network function virtualization (NFV), 5G mobile network,*

1. UVOD

U poslednjih tridesetak godina došlo je do intenzivnog razvoja više generacija mobilnih komunikacionih sistema, koji su sa sobom donosili novosti i poboljšanja u odnosu na prethodnu generaciju. Trenutno se kod nas i u svetu koristi četvrta generacija mobilnih komunikacionih sistema (4G) ili LTE (*Long Term Evolution*). Međutim, već se uveliko razvija novi mobilni komunikacioni sistem pete generacije. Paketski audio i video striming su sve popularnije usluge, To zahtijeva sve veće brzine prenosa podataka i manja kašnjenja. 5G sistemi se projektuju da zadovolje te zahtjeve a donese i nove funkcije.

Savremeni tempo života nužno donosi automatizaciju u gotovo svim aspektima života i rada ljudi. Osim toga, sve veći broj uređaja se povezuje u mrežu, i gradi koncept Internet stvari, IoT (*Internet of Things*).

Ovo je zapravo mreža uređaja koja ima zadatak da omogući razvoj takozvanih „pametnih kuća", a nakon toga i „pametnih gradova", „pametnih vozila" i slično. Da bi ovo bilo ostvareno, potrebno je implementirati novu, petu generaciju mobilnih komunikacionih sistema (5G). 5G je uveliko u fazi istraživanja i razvoja, a neke države su i implementirale 5G sisteme na nekim područjima.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Željen Trpovski, vanr. prof.

Tehnologije potrebne za njenu implementaciju svakodnevno se usavršavaju, kako bi pružile adekvatan kvalitet usluge. Četvrta generacija mobilnih komunikacionih sistema već ima zavidan kapacitet, brzinu i mala kašnjenja, ali se od pete generacije očekuje napredak u svim nabrojanim aspektima. Osim toga, od pete generacije se očekuje i manja potrošnja energije.

Softversko definisano umrežavanje (SDN) je nova tehnologija koja olakšava upravljanje i programabilnost mrežog sistema, što mrežu čini pouzdanijom centralizujući odvajanje upravljačke od ravni prosleđivanja podataka. Najvažnije prednosti SDN tehnologije ogledaju se u mogućnosti globalnog pregleda mreže, čime se omogućava centralizacija i prikupljanje podataka o mrežnom saobraćaju, a samim tim se poboljšava i kontrola nad mrežom. Centralizovanim pregledom mreže lakše se uočavaju greške i moguće opasnosti, pa je jednostavnije i brže pravovremeno reagovanje u cilju otklanjanja istih. Kod SDN tehnologije, pored prednosti, postoje i određeni bezbjednosni nedostaci.

SDN tehnologija olakšava DDoS (Distributed Denial of Service) i MITM (*Man In The Middle***)** napade, a mogu se očekivati napadi i na kontrolni sistem. Osim toga, s obzirom da je kontrola nad mrežom centralizovana, upad u sistem može kompromitovati cjelokupnu mrežu. Veliki bezbjednosni izazov predstavlja i očekivani ogromni broj korisnika, te će morati biti posvećena naročita pažnja u održanju funkcionalnosti i integriteta mreže.

Jedno od inovativnih rješenja za zaštitu od napada na kontrolni sistem bila bi replikacija lažnih kontrolera, čime bi se potencijalnom napadaču otežao pronalazak stvarnog kontrolera. Ovo nije savršeno rješenje, ali na ovaj način kontrolni sistem će barem donekle moći biti zaštićen. Ipak, potrebni su dodatni zaštitni sistemi, kako bi se osigurala bezbjednost SDN tehnologije. Paketi podataka kod SDN mreža se štite tehnikama kodiranja.

Međutim, kodiranje ne predstavlja zaštitu od distribuiranog uskraćivanja usluge, koje ima za cilj preopterećenje mrežnih resursa ogromnim brojem zahtjeva.

Rješenje za ovaj problem može biti uvođenje vremenskih markera, uz pomoć kojih bi mogao biti prepoznat DoS i DDoS napad [1].

2. PREDLOŽENA BEZBJEDNOSNA ARHITEKTURA 5G MOBILNIH MREŽA

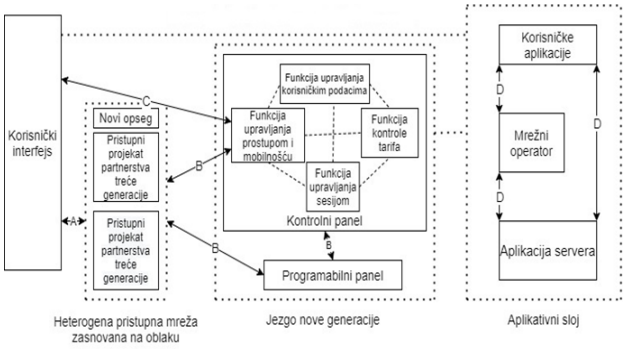
Bezbjednosna arhitektura mora biti realizovana tako da se prilagodi mrežnoj arhitekturi, ali mora da odgovori i na bezbjednosne izazove koje sa sobom nose 5G sistemi. Ona mora biti osmišljena tako da bude efikasna, ne opterećuje mrežne resurse i da bude adaptivna, kako bi se mogla prilagoditi trenutnim potrebama. Takođe, bezbjednosna arhitektura mora biti otporna na različite

prijetnje i akcije koje mogu kompromitovati bezbjednost podataka koji se šalju i primaju i korisnika koji komuniciraju.
Osim toga, mora se uzeti u obzir i činjenica da se od mobilnih mreža pete generacije očekuje da podrži veliki broj uređaja i korisnika. Zato se moraju osmisliti bezbjednosni mehanizmi koji će moći podržati ovako veliki broj uređaja i korisnika. Zbog tako velikog broja uređaja i korisnika, koji do sada nije viđen ni u jednoj mobilnoj mreži starije generacije, moraju se koristiti znatno napredniji bezbjednosni mehanizmi od ranije implementiranih. Svakako, treba se uzeti u obzir i potreba za izuzetno malim kašnjenjem zbog primene u sistemima za autonomnu vožnju, gdje reakcije u saobraćaju moraju biti brze i pravovremene.
S obzirom na ove zahtjeve, predlaže se razdvajanje ravni prosleđivanja podataka od kontrolne ravni, čime bi se omogućilo kodiranje podataka tako da budu fleksibilniji, da upravljanje mrežom i mrežnim saobraćajem postane jednostavnije, kao i da se ubrza proces provjere bezbjednosnih aspekata.

Najvažnije mrežne funkcije kontrolnog sloja jesu:

- Funkcija upravljanja pristupom i mobilnošću AMF (*Access and Mobility Management Function*). AMF nije uvijek neophodna, već zavisi od različitih primjena.
- Funkcija upravljanja sesijom, SMF (*Session Management Function*). Može postojati više SMF funkcija koje upravljaju različitim sesijama jednog korisnika za jedan AMF.
- Funkcija objedinjenog upravljanja podacima UDM (*Unified Data Management*). Ova funkcija takođe upravlja profilima za fiksni i mobilni pristup u 5G mreži.
- Funkcija kontrole politike PCF (*Policy Control Function*). Ovom funkcijom omogućava se upravljanje mobilnošću, kvalitetom usluge, romingom i sl. PCF funkcija vrši kontrolu AMF i SMF funkcija.

U 5G mrežama postoje četiri bezbjednosna domena, organizovana na sličan način kao u mobilnim mrežama starijih generacija.
Na slici 1. prikazana je bezbjednosna arhitektura sa četiri bezbjednosna domena, označenih sa A, B, C i D.



Slika 1. *Bezbjednosna arhitektura mobilne mreže [2]*

Bezbjednosni domen označen sa A, sačinjen je od bezbjednosnih funkcija koje obezbjeđuju bezbjedan pristup mreži, uz zaštitu od različitih vrsta napada. S obzirom na to da se u novom fizičkom sloju primjenjuju tehnologije kao HetNet, D2D (*Device-to-Device*), MIMO (*Multiple-Input and Multiple-Output*) i druge, pojavljuju se i problemi sa bezbjednošću koji se moraju sanirati i prevenirati.
Bezbjednosni domen označen sa B sačinjen je od funkcija čija je uloga zaštita od napada na klasični kablovski dio mreže, koji otežavaju bezbjednu razmjenu podataka. Bezbjednosni domen B nalazi se između RAN (*Radio Access Network*) i jezgra mreže nove generacije, kontrolnog i korisničkog sloja. Autentičnost, povjerljivost podataka i integritet predstavljaju najvažnije aspekte koji se provjeravaju u ovom bezbjednosnom domenu.
Bezbjednosni domen označen sa C sačinjen je od bezbjednosnih funkcija čija je uloga da obezbijede obostranu provjeru između jezgra mreže nove generacije i korisničkog interfejsa i to prije nego sam kontrolni sloj pristupi korisničkom interfejsu. Na ovaj način se vrši provjera autentičnosti.
Bezbjednosni domen označen sa D predstavlja bezbjednosne funkcije čija je uloga obezbjeđenje bezbjednosti poruka koje se razmjenjuju između korisničkog interfejsa i pružaoca usluge, ali i između korisnika i mrežnog operatora.

3. BEZBJEDNOSNI MEHANIZMI U 5G MOBILNOJ MREŽI

U cilju ostvarivanja bezbjednosnih zahtjeva 5G sistema neophodno je izvršiti unaprijeđenje postojećih bezbjednosnih mehanizama i iznaći nova rješenja koja će biti namjenski napravljena za primjenu u 5G sistemima. Inovativna arhitektura koju sa sobom donosi mobilna mreža pete generacije zahtijeva unaprijeđenje bezbjednosnih protokola, mali memorijski prostor i veoma brzu obradu podataka, u cilju smanjenja kašnjenja u procesu komunikacije. Osim toga, bezbjednosni mehanizmi moraju imati osobinu adaptivnosti, kako bi se mogli prilagoditi različitim vrstama napada i drugim bezbjednosnim prijetnjama.
Neke od bezbjednosnih tehnika koje bi mogle unaprijediti nivo bezbjednosti u mobilnim komunikacionim sistemima pete generacije jesu:

- provjera autentičnosti,
- održavanje povjerljivosti komunikacije,
- obezbjeđenje dostupnosti usluga i
- zadržavanje integriteta.

3.1. Provjera autentičnosti

Provjera autentičnosti se može vršiti na dva načina i to provjerom osobe i provjerom poruke. Provjera osobe ima za cilj da nesumnjivo utvrdi autentičnost strana u komunikaciji, dok s druge strane provjera poruke ima za cilj provjeru autentičnosti samog sadržaja komunikacije.
Postoje tri načina na koje se može izvršiti provjera autentičnosti. To su:

- provjerom same mreže,
- provjerom pružaoca usluge i
- provjerom i mreže i pružaoca usluge [6].

Kod SDN mreža preporučuje se brza provjera autentičnosti, koja ne upotrebljava kriptografske algoritme (koji neminovno oduzimaju vrijeme za provjeru autentičnosti), u cilju povećanja efikasnosti kod velikog broja zahtjeva. U poređenju sa digitalnom kriptografijom,

ovaj metod je teže potpuno kompromitovati. Metod se realizuje u više bezbjednosnih slojeva, pa je i stepen bezbjednosti veći [7].

3.2. Povjerljivost

Dva su aspekta koja čine povjerljivu komunikaciju, a to su povjerljivost podataka i privatnost. Povjerljivost podataka štiti podatke u toku prenosa od različitih vrsta pasivnih napada na način da ograničava pristup podacima samo ovlašćenim korisnicima. Privatnost s druge strane utiče na sprečavanje kontrolisanja i uticanja na samu informaciju, odnosno sprečava se pristup analitičkim podacima mogućim napadačima.

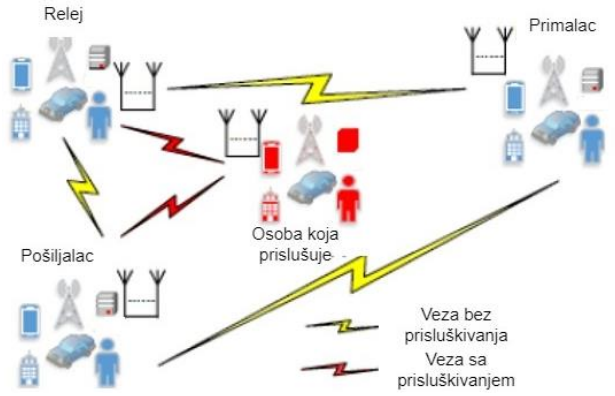
Za zaštitu povjerljivosti podataka i privatnosti najčešće se koristi metoda kodiranja. Metod kodiranja sprečava neovlašćen pristup i upotrebu podataka. Za kodiranje poruka veoma često se koristi metod simetričnog ključa. Princip simetričnog ključa zasniva se na tome da svaka strana u komunikaciji ima privatni ključ, pomoću kojeg se kodira i dekodira poruka. Privatni ključ mora biti distribuiran na bezbjedan način, kako bi se spriječilo da komunikacija bude kompromitovana. Metod simetričnog ključa je bezbjedan za slučajeve napada gdje napadač ima limitirane računarske resurse, pa ne može probiti šifru.

Da bi bio postignut visok nivo povjerljivosti podataka koji se u toku komunikacije razmjenjuju, potrebno je koristiti sledeće metode:

- upravljanje snagom,
- sigurnosni relej,
- vještački šum,
- obrada signala.

Upravljanje snagom signala je neophodno kako bi se otežala ili onemogućila rekonstrukcija komunikacionog signala i kako bi se na taj način izbjeglo presretanje i prisluškivanje komunikacije za neovlašćene osobe. Snaga signala prilagođava se smjeru komunikacije, sa ciljem da signal nosilac podataka dođe na odredište, odnosno do primaoca kojem je i namijenjen. Na ovaj način može znatno biti smanjena količina podataka koji su dostupni neovlašćenim licima.

Sigurnosni relej može biti posrednik u komunikaciji između dvije osobe. Sigurnosni relej pomaže pošiljaocu da bezbjedno prenese podatke do primaoca, kao što je to prikazano na slici 2.



Slika 2. *Princip rada sigurnosnog releja kao mehanizma zaštite poslanih poketa* [8]

3.3. Dostupnost

Pojam dostupnosti može se definisati kao stepen pristupačnosti usluge korisnicima, kao i u kom fizičkom dijelu teritorije je usluga pristupačna korisnicima. Dostupnost se može smatrati mjerom robustnosti sistema u slučaju napada. DoS i DDoS napadi predstavljaju zapravo napade na dostupnost usluge, a oni mogu onemogućiti korišćenje usluga koje pruža mrežni operator. Osim navedenog, ometanje takođe negativno utiče na dostupnost mreže, otežavajući komunikaciju. U 5G mrežama postojaće veliki broj čvorova, koji su potrebni za IoT, a to predstavlja veliki izazov u sprečavanju ometanja i DDoS napada, kako bi se obezbijedila dostupnost mreže [7].

3.4. Integritet

Bez obzira na činjenicu da se u svakoj komunikaciji vrši provjera autentičnosti poruke, čiji je zadatak provjeriti legitimitet izvora poruke, potrebno je takođe provjeriti da li je poruka u procesu prenosa izmijenjena ili duplifikovana. Zbog toga je od velikog značaja koristiti dodatni bezbjednosni mehanizam, čija je funkcija provjera integriteta poruke. Zadatak ovakvog mehanizma je da spriječi nepoželjno umnožavanje poruke, čime se zagušuje komunikacioni kanal, odnosno sistem, ali i da spriječi aktivne napade na komunikaciju, koje imaju za cilj da izmijene poslatu poruku. Apsolutan integritet poruke obezbjeđuje se na način da se obavlja obostrana provjera uz pomoć ključeva namijenjenih za provjeru integriteta.

4. ZAKLJUČAK ISTRAŽIVANJA NA OSNOVU KONTRAPRIMJERA RADI „OBARANJA“ HIPOTEZE

Ovaj rad je proizašao iz master rada, te su analizirane bezbjednosne prijetnje kao i kontraprimjeri obaranja polazne hipoteze za 5G sisteme. Na osnovu njih ponuđena su rješenja za zaštitu komunikacije u ovim sistemima. Kao prvi kontraprimjer se navodi aktivno prisluškivanje koje na osnovu povećanja broja antena (M>50) unutar bazne stanice ugrožava kapacitet tajnosti korisnika.

Dakle, aktivnom prisluškivaču kapacitet tajnosti je porastao i samim tim ugrozio legalnog korisnika. Drugi kontraprimjer se odnosi na distubuirani napad uskraćivanja usluge, gdje je jasno prikazano da se sav saobraćaj obustavlja, jer je došlo do zagušenja servera, a to se desilo nakon 3 sata od početka simulacije. Treći, ujedno i poslednji kontraprimjer se odnosi na slučaj „čovjek u sredini” (MITM), gdje „napadač” na osnovu URL adrese pristupa istom serveru kao i „žrtva” te uspijeva da preuzme odgovarajuće informacije.

preuzete informacije su napadaču neophodne za uspješan napad, uz pomoć unaprijed određenog pseudokoda.

Ipak, za obezbjeđenje maksimalne sigurnosti komunikacije pored navedenih kontraprimjera za obaranje hipoteze, biće potrebno implementirati tehnologije mobilnih mreža pete generacije u većoj mjeri – kao mreže sa velikim brojem korisnika.

Tokom implementacije vjerovatno će biti uočene i prijetnje i nedostaci, koje nije moguće uočiti prije samog realnog korišćenja. U ovom slučaju, adaptivni bezbjednosni mehanizmi mogli bi igrati ključnu ulogu za bezbjednu komunikaciju.

5. ZAKLJUČAK

Iz prethodnog izlaganja vidljivo je da svaka nova generacija mobilnih komunikacionih sistema nastaje kao rezultat potrebe za unapriđeđenjem postojećih mobilnih komuni-

kacionih sistema. Stoga je peta generacija mobilnih komunikacionih sistema nastala kao posljedica potrebe za razvojem mobilnih komunikacionih sistema četvrte generacije. Međutim, ona nije samo jednostavno unaprijeđenje LTE sistema, već je u svojoj osnovi bazirana na drugačijoj, naprednijoj arhitekturi i namijenjena je za znatno drugačije potrebe korisnika.
Mobilni komunikacioni sistemi pete generacije trebalo bi da u velikoj mjeri:

- ubrzaju prenos podataka,
- kašnjenja smanje na najmanju fizički moguću mjeru,
- pruže velikom broju korisnika na malom fizičkom području pristup širokopojasnoj Internet vezi,
- unaprijede pokrivenost signalom i
- unaprijede energetsku efikasnost sistema.

Da bi svi ovi, često kontradiktorni zahtjevi, bili mogući, potrebno je razviti tehnologije koje će razdvojiti hardverske i softverske funkcije i omogućiti virtualizaciju mrežnih funkcija.
Na taj način mobilne mreže pete generacije postaju daleko fleksibilnije i efikasnije, jer će postati moguće dijeliti mrežu na manje logičke cjeline, koje će biti konfigurisane u zavisnosti od potreba korisnika. Jezgro mreže, koje je u mobilnim komunikacionim sistemima starije generacije bila jedinstvena logička cjelina, u 5G mrežama moći će biti podijeljena na više jezgara, koja će obavljati funkcije bliže korisnicima, čime će se kašnjenje svesti na minimum.
Može se zaključiti da će mobilni komunikacioni sistemi pete generacije u prvih pet godina aktivnog rada imati ogroman broj korisnika, čime će se visoka ulaganja u razvoj istih isplatiti.
Uzevši u obzir naprijed navedeno, jasno je da su bezbjednosni zahtjevi 5G sistema znatno veći nego kod bilo kojeg drugog mobilnog komunikacionog sistema starije generacije.
Bezbjednost dakle, predstavlja izuzetno važno pitanje za mobilne komunikacione sisteme pete generacije. Ipak, imajući u vidu uspjeh ranijih generacija mobilnih komunikacionih sistema, može se očekivati da će razvojni timovi 5G sistema uspješno odgovoriti na sve izazove i projektovati sisteme bezbjedne, brze i pouzdane za krajnje korisnike.

6. LITERATURA

[1] „What is NFV: network functions virtualization basics“, dostupno na: https://www.electronics-notes.com/articles/connectivity/nfv-network-functionsvirtualisation/what-is-nfv-basics.php

[2] Eftychia Datsika : Radio resource management techniques for QoS provision in 5G networks PhD thesis dissertation, Universitat Politècnica de Catalunya (UPC), Barcelona, 2018

[3] Fernando Rodriguez, Ugo Dias, Divanilson Campelo, Robson Albuquerque, Se-Jung Lim and Luis Villalba, (2019): QoS Management and Flexible Traffic Detection Architecture for 5G Mobile Networks

[4] A., Kumar, Y., Liu, J. Sengupta: Evolution of Mobile Wireless Communication Networks: 1G to 4G, 2010

[5] Konstantinos Liolis, Alexander Geurtz, Ray Sperber, Detlef Schulz, Simon Watts, Georgia Poziopoulou, Barry Evans, Ning Wang, Oriol Vidal , Boris Tiomela Jou , Michael Fitch (2019): Use cases and scenarios of 5G integrated satellite-terrestrial networks for enhanced mobile broadband: The SaT5G approach, Journal of satellite communications and networking

[6] “5G Security: Forward Thinking Huawei White Paper”, HUAWEI WHITE PAPER, 2015.

[7] Dongfeng Fang, Yi Qian, Rose Qingyang Hu, „Security for 5G Mobile Wireless Networks“, IEEE, August 2017.,str. 5 - 6. [15] Dongfeng Fang, Yi Qian, Rose Qingyang Hu, „Security for 5G Mobile Wireless Networks“, IEEE, August 2017., Figure 12., str. 15

[8] [8] Robert W. Heath Jr., „Heterogeneous Networks“ , University of Texas at Austi

[9] Ankit Nilesh Ganatra (2017): Developments of 5G Technology, Master of Science Thesis, Governors State University

[10] Manuel Sainz (2015): 5G Techniques - Proof-of-concept Testbed , Master of Science Thesis, Aallborg University

[11] Konpal Shaukat Ali (2018): Modeling, Analysis, and Design of 5G Networks using Stochastic Geometry, King Abdullah University of Science and Technology Thuwal, Kingdom of Saudi Arabia

Kratka biografija:

Jovan Gojić, rođen je u Doboju 1994. god. Osnovnu i srednju školu završio u Doboju, gdje je završio i I cuklus studija na Saobraćajnom fakultetu. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnika i računarstva – Telekomunikacioni sistemi odbranio je 2021. god.
Kontakt: jgojic950@hotmail.com

Željen Trpovski rođen je u Rijeci 1957. godine. Doktorirao je na Fakultetu tehničkih nauka 1998. god. Oblast interesovanja su telekomunikacije i obrada signala.

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.

Zahvalnica:
Izradu ovog rada pomogao je Fakultet tehničkih nauka u Novom Sadu, Departman za energetiku elektroniku i telekomunikacije, u okviru projekta pod nazivom: "Istraživanja u oblasti energetike, elektronike, telekomunikacija i primenjenih informacionih sistema u cilju modernizacije studijskih programa".

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/15BE27Kljajevic

RAZVOJ RAČUNARSKE IGRE ČIJI JE CILJ SPASENJE TRADICIONALNOG ANTAGONISTE

DEVELOPMENT OF A COMPUTER GAME AIMED AT ASVING THE TRADITIONAL ANTAGONIST

Božidar Kljajević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Zadatak projekta bio je razviti računarsku igru od samog početka, odnosno osmišljanja ideje pa sve do krajnje realizacije računarske igre. Za realizaciju projekta krišćeni su Uniti (Unity), Blender (Blender) i Photoshop (Photoshop). Za potrebe igre osmišljene su pozadinska priča i priča igre, koje prate dašavanja u igri. Ciljevi igre su pomoći smrti, tradicionalnom antagonisti, i preživeti u ostatku igre. Kao rezultat dobijena je potpuno realizovana avantura kao zanimljiva računarska igra.*

Ključne reči: *Razvoj video igre, dizajn, modelovanje, priča, nivoi*

Abstract – *The task of the project was to develop a computer game from the very beginning, ie from the conception of the idea until the final realization of the computer game. Unity, Blender and Photoshop were named for the realization of the project. The background story and the story of the game are designed for the needs of the game, which follow the events in the game. The goals of the game are to help death, traditional antagonists, and survive in the rest of the game. The result is a fully realized adventure as an interesting computer game.*

Keywords: *Video game development, design, modeling, story, levels*

1. UVOD

Projekat koji se realizuje predstavlja video igru, čiji se razvoj prati od samog početka, od predstavljanja ideje, preko priče, razvoja i problema svih pratećih delova igre, do konačne verzije. Ideja je potekla od želje da se spoji više elemenata različitih žanrova objedinjenih u avanturu. Naziv projekta odnosno igre je From Beyond, što znači Sa one strane. Za realizaciju igre korišćeni su Unity, Blender i Photoshop.

Smrt je zarobljena. Da li si spreman da kreneš u avanturu? Vođen legendarnom pričom ulaziš u mračne prostorije sa ciljem da pronađeš smrt i da je oslobodiš kako bi stvari ponovo normalno funkcionisale. Kako se suočiti sa smrću i kako je osloboditi? Istražuj, suoči se sa strahovima i pobedi.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Ivetić, red. prof.

From Beyond je žanrovski određena kao avantura, ali u sebi krije i elemente drugih žanrova. Avantura uključuje istraživanje, rešavanje zagonetki, prikupljanje, dekodiranje poruka i kretanje lavirintima. Igra je third person realizovana u 3D-u i prvenstveno razvijana za PC. Igra prati Iroasa, glavnog junaka, koji je spletom okolnosti izabran da krene u avanturu u mračni svet sa ciljem da oslobodi Thanatosa, odnosno smrt, kako bi svet ponovo normlano funkcionisao. Cilj u igri je istražiti, rešiti i sakupiti potrebno. Glavni cilj je osloboditi zarobljenu smrt. Smrt se osloađa kada se pronađe lobanja i donese do smrti. Ono što igru čini jedinstvenom je svakako zabavan gameplay, ali i kako je već napomenuto jedinstvena kombinacija više elemenata drugih žanrova u okviru avanture. Kombinacija horora i mozgalica uklopljenih u jednu igru. Originalna pozadinska priča (eng. backstory) data je u igri i nadograđena fantastičnom pričom same igre. Priča ima zanimljiva dešavanja i fantastično prati sva dešavanja u igri. Ono što predstavlja jedinstvenost igre je činjenica da je u igri potrebno osloboditi smrt. Dok je u igrama standardno pravilo boriti se protiv smrti ili je smrt predstavljena kao nešto nepoželjno, u igri From Beyond cilj je pomoći smrti i osloboditi je.

From beyond je namenjen ljudima širom sveta, bez geografskih ograničenja ili tendencija na nekom delu sveta. Sa druge strane, što se tiče starosne granice, igra je namenjena tinejdžerima starijim od 12 godina i odraslim osobama. Namenjena je avanturistima, osobama koje vole horor, misteriju, fantastiku, zagonetke i istraživanje. Cilj je da se kod igrača izazove osećaj uzbuđenja, ali i straha, čemu doprinose ambijent i muzika u igri. Takođe kod korisnika se izaziva ushićenje, odnosno želja za napredovanjem. Kad otkrije ili reši neki deo, kod korisnika se formira uzbuđenje i želja da vidi šta je sledeće, te nastavlja da igra. Od igrača se očekuje i razmišljanje i rešavanje zagodnetki koje nisu jednostavne, a to je još jedan od razloga zašto je granica za igrača postavljena na minimum 12 godina. Estetika je prisutna u igri i ima za cilj da izazove estetsku zahvalnost igrača, ali pored estetskog momenta neki predmeti koji doprinose estetici kriju i značajne informacije u igri.

2. NARATIVNI DIZAJN

Ono što treba da pokrije jedan dobro odrađen narativni deo igre nije samo glavna priča koja prati igru, već i dešavanja pre ili eventualno posle glavne radnje. Radnja, odnosno priča koja nosi glavni teret svakako jeste onaj deo koji je prisutan u radnji igre, ali podjednako je bitno

dati i uvid u stvari kako se došlo do radnje u igri, odnosno predstaviti širu sliku ili dati pozadinsku priču koja objašnjava glavna dešavanja, tačnije glavne probleme u samoj radnji igre.

2.1. Pozadinska priča

Pre mnogo godina, vođeni idejom o životu bez smrti, četiri učena čoveka tog vremena, među kojima je bio jedan slikar, okupili su se u mračnim i zabačenim prostorijama sa idejom da smisle plan kako da zarobe smrt. Smrtnost je deo čovekove priode, ali oni su hteli to da promene i da ljudi postanu besmrtni. Izvodili su eksperimente opasne po zivot, kako bi namamili smrt da dodje ka njima. Smrt je došla i upala u već pripemljenu i osmišljenu zamku. Oduzeli su joj lobanju kako ne bi mogla da izađe i na taj način smrt ostaje zarobljena. Slikar uviđa da prave grešku, ali njegovi saradnici ne žele da ga slušaju. Smrt ostaje zarobljena i od tog momenta ljudi žive večno, bez umiranja. Prirodni proces umiranja je narušen, ljudi žive sa opakim bolestima koje bi se u prirodi stvari završavale smrću, ali sada smrti nema i ljudi žive sa bolestima i muče se.

2.2. Priča igre

Priča počinje sa prikazom problema u stvarnom svetu junaka. Ljudi dugo žive, nema umiranja, moge bolesti ne završavaju se smrću, već ljudi žive sa tim opakim bolestima i muče se. Nije to život, ali smrti nema, prirodni proces je narušen već duže vreme. Iroas saznaje za legendu o smrti, odnosno kako je smrti oduzeta lobanja i kako je zarobljena. Zaintrigiran je legendom, ali nema dovoljno hrabrosti da se upusti u avanturu. Ne želi da napusti porodicu i bliske ljude, jer neki od njih su bolesni. Takođe u Iroasovoj glavi se ponavlja pitanje "Da li je smrt na našoj strani, da li je pametno osloboditi je?". Čarobnjak Magos daje Iroasu datalje o legendi i otkriva mu destinaciju gde mora poći u nameri da oslobodi smrt. Iroas prihvata izazov i odlazi na mračno mesto gde smrt čeka zarobljena. Iroas mora rešiti testove i zagonetke kako bi prešao prepreke koje su stavljene ispred njega. Susreće se sa Thanatosom - smrt i shvata da nema povratka nazad. Thanatos daje Iroasu zadatak da pronađe lobanju i da mu je donese. Na tom putu ga čeka još testova, koje ako ne reši i ne donese lobanju, ostaje zarobljen sa Thanatosom u mračnim prostorijama. U skrivenoj prostoriji konačno pronalazi lobanju koju traži. Thanatos dobija lobanju i konačno je slobodan. Pokazuje Iroasu put, ali taj put nije bezbedan. Kako bi pobegao sa ovog mračnog mesta pred Iroasom su nova iskušenja i ne sme praviti greške sada kada je smrt slobodna i motri na njega. Kako je Iroas oslobodio Thanatosa, prirodni proces umiranja je ponovo uspostavljen.

3. KARAKTERI

U svakoj video igri bitan deo priče i igre su karakteri. Igrači se često poistovete sa glavnim karakterom ili im pažnju privlači neki od sporednih karaktera. Kao glavni nosioci radnje karakteri moraju biti autentični i adekvatno usklađeni sa pričom i ambijentom radnje. Na sceni u video igri prisutna su tri karaktera, dok se u priči pominju još neki. U igri su prisutni glavni lik Iroas, Thanatos - karkater koji predstavlja smrt i Magos – čarobnjak, odnosno mentor i pomoćnik u igri.

3.1. Modelovanje i animiranje

Prvi korak u realizaciji je samo modelovanje karaktera. Kako se modeluje karakter čija je namena da se koristi u video igri, onda se model pravi tako da bude *Low Poly* model. Glavna ideja je da model bude sačinjen od manjeg broja poligona kako bi model zahtevao manje računarske snage za renderovanje. Stoga se *Low Poly* modeli i koriste u video igrama i sistemima virtuelne realnosti.

Ovako napravljen model se dalje priprema za dodavanje teksture. Model se odmota (eng. *unwrap*), odnosno ceo model se prikaze na 2d površini kako bi se na njega primenila odgovarajuća tekstura. Da bi se model odmotao, na modelu su ručno pravljeni rezovi (eng. *seams*) kako bi model mogao da se prikaže u 2d-u. Rezovi su pravljeni tako da se uglavnom nalaze na ivicama modela, kada je to bilo moguće, ili na delovima tako da rezovi budu što manje upečatljivi i vidljivi. Nakon određenih rezova model se transformiše u UV mapu, odnosno 2d delove modela prikazane u ravni. Na tako razvijen model u 2d potrebno je primeniti teksturu. Kako bi se primenila tekstura kao jedna slika, ručno je pravljena takva slika, delom u Blenderu, delom u Photoshopu. Nakon primene teksture na model, sledeći korak u realizaciji funkcionalnog modela je rigovanje, odnosno ubacivanje skeleta. Za pravljenej skeleta iskorišćena je opcija koju nudi Blender, a to je *Basic Human Meta-Rig*. Primenom ove opcije dobija se skelet, nakon čega se kosti pažljivo i precizno postave na odgovarajuće pozicije tako da se poklapaju sa prethodno kreiranim karakterom. Nakon preciznog postavljanaj kostiju na odgovarajuće pozicije, koristi se *Generate rig*, postojeća opcija u Blenderu koja omogućava povezivanje kostiju i kontrolu na ispravan način. Nakon što je skelet napravljen potrebno je povezati karaktera i kosti u procesu koji se zove *skinning*. Model je *parent*-ovan na kosti sa *automatic weighs*, a zatim u *weight paint* modu su dorađene težine gde je bilo potrebno. Na ovaj način svaka tačka modela povezana je određenim procentom za odgovarajuće kosti. Ovako odrađen model spreman je dalje za animiranje. Odrađene su tri animacije: idle, hod i skok. Idle je jednostavna animacija sa animiranjem tri kontrole tako da se stekne utisak da karakter diše. Animacija traje 34 frejma i prvi i poslednji frejm su isti kako bi se mogao efikasno uključiti loop animacije. Animacija hoda je takođe loop animacija gde prvi i poslednji frejm izgledaju isto, s tim da u sredini postoji frejm koji je suprotan od početnog i krajnjeg, odnosno suprotne ruke i noge se nalaze napred i nazad. Urađene su ključne poze: contact, down, pass, up. Treća animacija je animacija skoka koja se sastoji iz tri animacije, kako bi posle mogle odvojeno biti implementirane u igri. Prva animacija je odskok, koja prelazi u animaciju inAir, odnosno animaciju padanja, koja ide u loop sve dok karakter ne treba da se dočeka, a tada ide treći deo, odnosno doskok.

3.2. Sinopis likova

Poznati psiholog Karl Jung objašnjava razlog za pripovedanje u smislu nesvesnog znanja sa kojim smo svi rođeni, a opet možemo da ga nikada ne budemo direktno svesni. U okviru ovog znanja nalaze se univerzalne teme i arhitipovi koji se pojavljuju u obliku priča i tipova likova u umetnosti, muzici, književnosti, filmu i igrama.

Arhetipovi se koriste u svim zabavnim medijima kako bi pojačali vezu publike sa pričom. Arhetipovi po Jungu prisutni u igri From Beyond: heroj, mentor, čubar i senka.

Heroj - suočava se sa problemom priče, sprovodi radnju igre, preuzima rizik i odgovornost. Sinopis lika - Ime: Iroas (gre. *ήρωας*) na grčkom jeziku junak – slika 1. Opšti tip: fantastični, stvoren samo za potrebe igre. Karakter je muškog pola, starosti između 25 i 30 godina. Fizički izgled: karakter prosečne građe, izgledom predstavljen kao prosečan čovek iz naroda, akcenat je na njegovoj moralnoj veličini. Vizuelne karakteristike – četiri boje (crvena, crna, zelena i siva). Na jakni oslikan zmaj (hrabrost, odvaţnost). Na glavi šlem kao unikatan deo karaktera – sigurnost. Razvoj lika – Ide na putovanje da reši problem koji je zadesio ne samo njega, već i celokupno čovečanstvo. Moralnu snagu pokazuje kada problem svih ljudi stavlja u prvi plan, gde pokazuje brigu ne samo za sebe već i za zajednicu. Razvoj lika se ogleda u njegovoj bojazni da krene na put, hrabrosti da ipak ode u avanturu i suoči se sa izazovima, preko susreta sa smrću, i konačnog povratka gde je junak sad bogatiji za jedno veliko iskustvo. Pokazuje da može da se nosi sa strahom i legendom o smrti. Pokret – ciklus kretanja i idle prikazuju da je heroj hrabar i siguran u sebe, iako se nalazi na opasnom mestu i na opasnoj misiji, ipak pravi brze i odvažne pokrete i užurbano se kreće ka opasnosti.

Slika 1. – *Iroas – glavni junak*

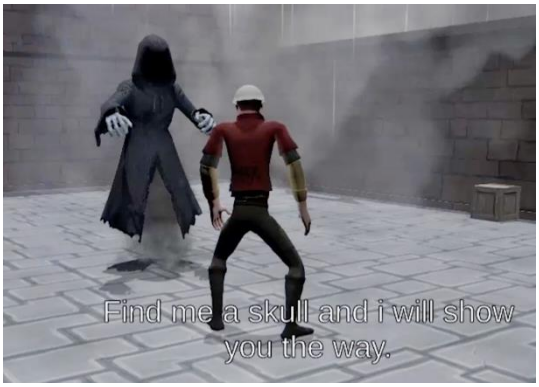
Mentor - stariji savetnički lik, oslonac heroja u avanturi. Ime: Magos (gre. *mágos*), što na grčkom jeziku znači čarobnjak. Opšti tip: mitski, u mitskim pričama često se pominju čarobnjaci. Lik je muškog pola, starosti oko 400 godina. Fizički izgled: visine iste kao heroj, seda i duga kosa, seda brada. Vizuelne karakteristike – dominira siva boja, sivo odelo, plašt, šešir i štap. Osobine – čarobnjak je mudar, živi dovoljno dugo da poznaje događaje koji čekaju heroja. Simboli mudrosti kao kod svakog čarobnjaka su dugačka brada, šešir i štap. Čarobnjak poznaje legendu o smrti, daje dovoljno informacija samom heroju. Vokalne karakteristike: dubok i smiren glas.

Čuvar – heroju blokira napredovanje junaka dok junak ne dokaţe svoju vrednost, u ovom slučaju dok ne donese potreban predmet. Sinopis lika: Ime: Thanatos (gre. *θάνατος*), na grčkom jeziku znači smrt. Opšti tip: mitski, smrt se često pominje u mitovima i legendama. Lik je muškog pola, besmrtan. Fizički izgled: lik smrti je viši malo u odnosu na heroja, krupniji je od heroja. Vizuelne karakteristike – dominira crna boja, crni plašt. Na rukama ima duže nokte nalik na kandže. Iskrzali i pocepani krajevi odeće. Osobine – nestrpljiva, željna slobode, takože mudra. Pokret – Idle animacija oslikava neki vid tapkanja u mestu, odnosno bespomoćnost i nestrpljenje smrti. Smrt se takođe nestrpljivo kreće u krug po prostoriji, dok se ne susretne sa Iroasom, kada prelazi u Idle.

Senka – četiri vidjenija čoveka tog vremena. Razmišljaju suptorno od heroja. Odgovorni su za problem koji je zadesio heroja. Nisu u potpunosti zli, već imaju suprotan stav od heroja po pitanju smrti. Pohlepno su postupili bez razmišljanja o posledicama i tako naneli zlo svima.

4. GAMEPLAY

Na početku igre Iroas je u prostoriji iz koje vode samo jedna vrata. Vrata su zaključana i mogu se otvoriti samo unošenjem tačne lozinke. U istoj prostoriji nalazi se slika koja trenunto Iroasu ne može pomoći i namenjena je za nastavak igre. Na jednom od zidova nalazi se kvadrat koji se rotira i koji krije šifru od vrata. Nedostajuća mesta na kvadratu koja su obeležena zvezdicom predstavljaju šifru za vrata. U pitanju je niz brojeva od 85 do 92. Ovaj niz je odabran jer cifre koje nisu pod zvezdicom su 0, 6, 8 i 9, a kako se kvadrat rotira ove cifre imaju smisla i rotirane. Kada Iroas priđe šifratoru od vrata i Igrač pritisne komandu E otvara se šifrator i moguće je uneti šifru. Nakon otključavanja i prolaska kroz vrata Iroas zatiče Thanatosa u kružnom užurbanom hodu. Kada Iroas priđe Thanatosu on se zaustavi – slika 2. Pri susretu sa Thanatosom, glavni izazov Iroasu daje baš on. Thanatos daje Iroasu zadatak da pronađe lobanju koja je sakrivena. Ne daje mu previše informacija, ali obećava pomoć u nastavku ako mu donese lobanju.



Slika 2. – *Susret sa Thanatosom*

Sada se pred Iroasom nalazi dvoje vrata. Jedna vrata zahtevaju ključ kako bi se otvorila a druga unos tačne šifre. Kako trenutno ključ nije moguće naći, igrač se usresređuje na vrata koja zatevaju lozinku. Iznad šifratora se nalazi kruna sa četiri kruga na vrhu, a pored umetničkih dela na zidovima takođe se nalaze krune ali sa po jednim krugom koji označava redosled lozinke. Ovo označava da se na slikama krije lozinka za otvaranje ovih vrata. I dalje je primarni cilj pronaći lobanju. Otključavanjem vrata čiju su šifru krile slike, Iroas ulazi u novu prostoriju u kojoj se ne nalazi lobanja, ali koja naizgled nema izlaz i ne vodi nigde. Još jednom umetničke slike na zidovima imaju ključnu ulogu u otkrivanju tajne prostorije. Četiri protreta, gde svaki pokazuje prstom u smeru gde igrač treba da se kreće i gde je skrivena prostorija. Pored svake slike nalazi se simbol ruke sa uperenim kažiprstom, što predstavlja pomoć

igraču da obrati pažnju na šta portreti sa slike pokazuju. Kada igrač isprati putokaze koji pokazuju slike nailazi na zid koji izgleda kao ćorsokak, ali kada se igrač dovoljno približi zidu vidi se prolaz na zidu, kroz koji igrač može da prođe. Iza prolaza je skrivena prostorija gde se nalazi lobanja koju Iroas traži. Kada se pokupi lobanja pritiskom komande E, potrebno je istim putem vratiti se do Thanatosa i doneti mu lobanju. Thanatos kao što je i obećao otvara novi put za Iroasa. Novi put čine plaforme kojima se Iroas mora kretati kako bi uspeo da pobegne sa ovog mračnog mesta. Put započinje sa nekoliko platformi u vidu stepenika, zatim prelazak preko uske grede. Nakon grede Iroas mora da skoči na platfomu koja se kreće napred nazad nakon koje sledi još jedna uska greda. Sledeći izazov za Iroasa predstavljaju platforme koje padaju čim stane na njim, tako da se preko njih mora brzo kretati i na njima se ne zadržava.

Naredna platforma se kreće na gore kada Iroas stane na nju i vodi ga direktno do portala kroz koji Iroas mora proći. Nakon prolaska kroz portal Iroas se nalazi u središtu olujnih oblaka i od pada i gubitka života ga dele samo platforme. Iroas u trenutku prolaska kroz portal ima dva života i sada kada je smrt slobodna mora pažljivo da se kreće. Nakon prolaska kroz portal Iroas se susreće sa Magosom koji za igrača predstavlja sigurnu tačku, odnosno ako padne sa platformi biće vraćen na ovu poziciju ako mu je preostalo života. Pred Iroasom je naredna prepreka u vidu platformi gde neke platforme padaju, a neke ne, tako da nije siguran na kojoj platformi se sme zadržati, a na kojoj ne sme, stoga se mora brzo kretati. Nakon prelaska platformi koje padaju pred Iroasom se nalazi lavirint. Glavna karakteristika Lavirinta je da se sastoji od platformi i da se u odnosu na platformu na kojoj se stoji uvek pojavljuju jedna platforma napred i jedna nazad. Ovako pojavljivanje platformi usporava igrača da se polako i pažljivo kreće i unapred razmišlja gde bi trebalo da ide. Prva raskrsnica u lavirintu daje Iroasu mogućnost levog i desnog puta. Ako izabere levi put i nastavi njime da se kreće, ponovo će doći na raskršće, gde jedan put vodi ka prikupljanju dodatnog života, a drugi put vodi ka prikupljanju ključa. Nakon što prikupi neophodan ključ i ako je potrebno dodatni život, Iroas se vraća na početno raskršće i kreće na desnu stranu. Nakon kretanja ovim putem ponovo nailazi na raskršće, ovoga puta jedan put je pogrešan i vodi u ćorsokak, a drugi put vodi ka novom portalu. Nakon prolaska kroz ovaj portal Iroas biva vraćen na prethodno područje ispred vrata za koja je bio potreban ključ. Ukoliko igrač ne sakupi ključ mora istim putem da ide ponovo do lavirinta gde se ključ nalazi. Ako je ključ ipak pokupljen kada se priđe vratima koja zahtevaju ključ, ona će se otvoriti i Iroas će moći da napusti ovo mračno mesto. Prolaskom kroz poslednja vrata Iroas je uspešno rešio zadatke, vratio je lobanju smrti i na taj način smrt je slobodna, a zatim je sebe spasao i pobegao je sa tog strašnog mesta.

5. ZAKLJUČAK

Igra From Beyond uspešno je realizovana. Cilj je bio napraviti igru žanrovski određenu kao avantura, ali sa elementima i drugih žanrova. Uklapanjem elemenata drugih žanrova sa avanturom dobija se širi spektar potrncijalnih korisnika, odnosno igrača. Rezultat predstavlja kompletno formirana igra realizovana u četiri scene. Prva scena koja je realizovana da bude početna gde igrač može da oseti atmosferu koja ga očekuje u nastavku i odmah u prvoj sceni se upoznaje sa glavnim junacima igre. Druga scena je namenjana da se sazna šta su problemi u igri i kako su ti problemi nastali, odnosno pozadinska priča o nastanku sveta igre. Treća scena predstvalja srž igre, odnosno na njoj su smeštena sva dešavanja u igri. Ova scena krije sve zadatke i avanture namenjene korisnicima. Poslednja scena aktivira se po uspešnom rešavanju svih prepreka i zadataka i predstavlja potvrdu igraču da je uspešno prešao igru.

Glavnu autentičnost igre predstavlja jedan od ciljeva igre, tačnije u igri je određen zadatak da se oslobodi smrt. Smrt je u većini igara i priča prestavljena u liku antagoniste, ali u igri From Beyond cilj je pomoći smrti. Kao rezultat ovog nesvakidašnjeg cilja, igra postaje jedinstvena i drugačija.

Dobro usaglašen narativni dizajn i realizacija priče pretvorena u video igru bili su ključ za uspeh. Igra je razvijana prvenstveno za PC, ali postoji mogućnost njenog lakog prenosa i na ostale platforme. Adekvatna priča i realizacija mogu da posluže kao inspriacija za pravljenje stvarnog escape room-a sa zadacima i zagonetka realizovanim okviru stvarnog prostora. Danas je sve prisutnije da se na osnovu video igre prave filmovi i serije. Igra From Beyond sa svojom autentičnom pričom ima veliki potencijal da se u budućnosti na osnovu nje uradi film ili serija. Pored svih navedenih potencijala, najbitniji potencijal igre ogleda se u mogućnosti formiranja nastavka igre. Priča i igra su dovoljno intrignatne i otvorene, tako da postoji prostor formiranja nastavka kako priče tako i igre. Nastavak i budući projekat mogli bi da obuhvate dešavanja koja su nastala nakon oslobađanja Thanatosa. Mogući sled dodađaja mogao bi da uključi i ponovno udruživanje Iroasa i Thanatosa, ali ovaga puta Iroas bi bio negativac – obavljao bi prljav posao za Thanatosa. Nakon oslobađanja Thanatos je oduzeo život Iroasovoj voljenoj osobi, ali mu je obećao da će je vratiti ako se povinuje njegovim naredbama. Thanatos drži Iroasa u šaci i pravi od njega negativca. Priča bi pratila glavnog karaktera koji ne bi spasavao svet i radio u službi dobra, već bi igra pratila Iroasa koji je zaslepljen obećanjima Thanatosa postao negativca. Realizacija ove potencijalne ideje i igre, kao nastavka igre From Beyond, gde igra prati nekada pozitivnog lik, a sada negativca koji ima tendenciju da ponovo bude na strain dobra, predstavlja veliki potencijal. Ova ideja za nastavak igre otvara vrata budućim projektima i uspešnim nastavcima igre From Beyond.

Kratka biografija:

Božidar Kljajević rođen je u Novom Sadu 1997. god. Fakultet tehničkih nauka upisao je 2016 god. Na studijskom programu Računarstvo i Automatika. Bečelor rad odbranio je 2020. god. Master rad na usmerenju Multimedija i računarske igre odbranio je 2021. god. kontakt: bozidar.kljajevic07@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/15BE28Pantic

IZBOR HIPERPARAMETARA ALGORITAMA DUBOKOG UČENJA SA POTKREPLJENJEM PRIMENOM GENETSKOG ALGORITMA

SELECTION OF HYPERPARAMETERS OF DEEP REINFORCEMENT LEARNING ALGORITHMS USING A GENETIC ALGORITHM

Vasilije Pantić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Ovaj rad rešava problem hoda robota u prostoru pomoću algoritama dubokog učenja sa potkrepljenjem koji se optimizuju pomoću genetskog algoritma.*

Ključne reči: *Duboko učenje sa potkrepljenjem, genetski algoritam*

Abstract – *This paper solves the problem of robots walking in space using deep reinforcement learning algorithms that are optimized using a genetic algorithm.*

Keywords: *Deep Reinforcement Learning, Genetic Algorithm*

1. UVOD

Veštačka inteligencija (eng. *artificial intelligence - AI*) ima za cilj pravljenje programa koji bi imitirao ljudska ponašanja i rezonovanje. *AI* na osnovu prethodnog iskustva (podataka kojim ima pristup) pokušava da nađe šablon po kome su mapirani podaci tako da dolaskom novih podataka može da donosi odluku ili rezultat sličan tim podacima. *AI* zahteva ogromnu količinu podataka u odnosu na ljudski mozak, kome je potrebno nekoliko puta da ugleda dva različita objekta da zna da ih razlikuje, dok to sa *AI*-em nije slučaj.

Pored traženja šablona u podacima, postoji i način učenja donošenja odluka načinom kazni i nagrada. Kao što čovek teži u životu nagradama i pozitivnim stvarima, tako je i nad ovim pristupom cilj da se loše odluke penalizuju negativnom nagradom (kaznom), dok bi odluke koje vode ka željenim ciljevima davalo pozitivne nagrade.

Rad rešava problem kretanja robota (koji predstavlja repliku čoveka) sa ciljem da hodajući u prostoru održava ravnotežu i uspešno korača napred metodama nagrađivanja i kažnjavanja, izraženim kroz mašinsko učenje i pronaći optimalna rešenja u pogledu onih koja su najbolja i time postići maksimalne vrednosti i na najbolji način obučiti robota da hoda.

2. IMPLEMENTACIJA REŠENJA

2.1. Uvod u problem i tehnologije

Problem koji ovaj rad rešava pomoću algoritama dubokog učenja sa potkrepljenjem i genetskim algoritmom se tiče učenja robota hodu (eng. *humanoid robot walk*).

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Milan Rapaić, red. prof.

Okruženje koje se koristi za ovaj problem je predefinisan (nije pravljen od strane autora rada) i u pitanju je *OpenAI Gym* [4] i *PyBullet Gym* [5], koji daju gotovo okruženje u kome se robot nalazi, samog robota sa svojim akcijama koje može da izvršava, kao i sistem nagrađivanja i kažnjavanja.

Programski jezik iskorišten za implementaciju rešenja jeste *Python* sa radnim okvirom za duboko učenja *PyTorch* [6]. Na slici 1 dat je izgled robota i okruženja koji se rešava u ovom radu.

Slika 1. *Primer robota i okruženja problema [5]*

Cilj robota prilikom hoda jeste da se uspešno kreće u prostoru (prostor nema prepreke). Stanje prostora u kome se robot nalazi je predstavljeno kao niz veličine 44 popunjen vrednostima od -1 do 1 gde ti brojevi prezentuju ugao i poziciju zglobova robota. Kao akcije koje robot vrši u ovom prostoru, prezentovane su kao niz veličine 17 popunjen vrednostima od -1 do 1 gde ti brojevi predstavljaju momente sile primenjene na zglobove. Po pitanju nagrađivanja robota za uspešne akcije, robot ostvaruje nagradu onda kada se pomera unapred i kada protivi teži gravitacije.

Formalnije izraženo, cilj je pronaći politiku π kojom će agent postizati maksimalnu kumulativnu nagradu nad trajektorijom, tačnije da će povrat jedne epizode biti maksimalan. Robot počinje iz početnog stanja s_0 te je cilj izvršavati akcije koje će omogućiti robotu da ne završi u neželjenom terminalnom stanju, koje zapravo predstavlja njegov pad.

Da bi za ovo rešenje postojalo željeno terminalno stanje, određen je maskimalan broj akcija koje agent može izvršiti u jednoj epizodi, tako da se sada problem odnosno kriterijum definiše kao postizanje maskimalnog broja akcija koje agent može izvršiti.

Algoritmi koji se koriste za rešavanje ovog problema su *PPO* [3] i *TRPO* [1], dok se genetski algoritam koristi za optimizaciju hiperparametara nad algoritmima tako da se ostvari bolji i brži rezultat. Sav kod implementacije rešenja problema se nalazi na [7] i [8].

2.2. Implementacija neuronskih mreža

Oba algoritma dubokog učenja sa potkrepljenjem (*TRPO* i *PPO*) koriste neuronske mreže da aproksimiraju politiku i funkciju vrednosti stanja. Neuronska mreža koja aproksimira politiku naziva se glumac (eng. *actor*), a neuronska mreža koja aproksimira funkciju vrednosti stanja se naziva kritičar (eng. critic).

Ideja pristupa sa glumcem i kritičarem jeste da kritičar aproksimira funkciju vrednosti za celi prostor na osnovu nagrada koje agent ostvaruje vršeći trajektorije u datom prostoru odnosno na osnovu dosadašnjeg iskustva, dok je cilj glumca da ažurira parametre politike i aproksimira politiku agenta u pravcu sugerisanom od strane kritičara.

Obe arhitekture neuronskih mreža imaju isti ulaz u mrežu - niz veličine 44 koji prezentuje stanje u kome se agent trenutno nalazi.

Broj skrivenih slojeva za obe arhitekture je 2, sa brojem neurona po sloju 128 i 64 redom. Jedina razlika između ove dve arhitekture jeste u dimenziji izlaznog sloja, koji za glumca ima 17 neurona u izlaznom sloju da predstavi akciju koju treba agent da izvrši u datom stanju, dok kritičar ima samo jedan neuron u izlaznom sloju da predstavi funkciju vrednost tog stanja. Obe arhitekture kao aktivacionu funkciju (osim nad izlaznim slojem) koriste *ReLU* aktivacionu funkciju.

Kako je u pitanju kontinualan prostor akcija koje agent i izvršava i kako se očekuje da vrednosti vektora koji se prosleđuje kao akcija ima vrednosti u intervalu $[-1,1]$, tako se za izlaz glumca koji treba da proizvede sledeću akciju koristi *Gaussova MultivariateNormal* distribucija [9], koja se zapravo brine da izlaz iz neuronske mreže baš daje takve vrednosti, postavljajući srednju vrednost distribucije na izlaze neuronske mreže glumca.

2.3. Implementacija algoritama dubokog učenja sa potkrepljenjem

Hiperparametri koji su od interesa konkretno za *TRPO* algoritam [1] i koji su bitni za adekvatno treniranje samog algoritma na zadatom problemu su sledeći:

- *delta*: ograničenje KL-divergencije
- *gamma*: faktor obezvređivanja nagrada
- *cg_delta*: ograničenje za *conjugate gradient* algoritam [2]
- *cg_iterations*: broj maksimalnih iteracija za conjugate gradient algoritam [2]
- *alpha*: koeficijent za računanje pravilnog gradijenta da zadovolji *delta* koeficijent
- *backtrack_steps_num*: maskimalan broj koraka kojim se ide unazad za računanje pravilnog gradijenta da zadovolji *delta* koeficijent
- *critic_epoch_num*: broj epoha za treniranje kritičara za jednu epohu *TRPO* algoritma

Implementacija TRPO algoritma data je na [7]. Kod TRPO, neuronska mreža kritičar se ažurira gradijentnim spustom sa ciljem da estimira funkciju vrednosti stanja, dok neuronska mreža glumac računa manuelno gradijent kao maksimalni korak koji unapređuje politiku, a zadovoljava ograničenje dozvoljene razlike stare i nove politike, tačnije $delta$.

Hiperparametri koji su od interesa konkretno za *PPO* algoritam [3] i koji su bitni za adekvatno treniranje samog algoritma na zadatom problemu su sledeći:

- *number_of_network_updates_per_iteration*: broj epoha za treniranje kritičara i glumca za jednu epohu *PPO* algoritma
- *discounting_factor*: faktor obezvređivanja nagrada
- *clip_range*: specifično ograničenje koje ograničava koliko stara i nova politika mogu da se razlikuju

Implementacija PPO algoritma data je na [7]. Kod PPO, neuronska mreža kritičar se ažurira gradijentnim spustom sa ciljem da estimira funkciju vrednosti stanja, dok neuronska mreža glumac se ažurira gradijentnim usponom sa ciljem da estimira politiku agenta.

2.4. Implementacija genetskog algoritma

Ideja genetskog algoritma za problem jeste da nađe optimalne parametre za TRPO algoritam [1] (s obzirom da je to algoritam sa više parametara, te zahteva više eksperimentisanja i podešavanja), te je cilj pronaći kandidata koji postiže najbolje performanse u prvih nekoliko desetina epoha nad problemom.

Hiperparametri koji se tiču konkretno genetskog algoritma su sledeći:

- *number_of_agents*: broj kandidata po generaciji
- *number_of_generations*: maksimalan broj generacija koje će genetski algoritam da produkuje
- *number_of_iterations*: broj epoha TRPO algoritma koje će jedan kandidat da izvrši u generaciji
- *top_limit_agents*: broj elitnih kandidata za prenos u sledeću generaciju
- *mutation_chance*: šansa da će detetovi genomi biti mutirani

Kao fitnes funkcija svakog kandidata se gleda *rewards-to-go* koje je ostvario kroz *number_of_iterations* broja epoha, sa dodatkom da se prvo stanje (početno) postavlja na nulu jer je nasumično i ne igra ulogu.

Za inicijalizaciju populacije, ideja je nasumično dodeliti vrednosti parametara koje želimo da optimizujemo.

Za selekciju roditelja, ideja je da dva roditelja uvek kreiraju dva deteta, koja će imati drugačiji način ukrštanja, sa ciljem razlikovanja među decom. Svaki roditelj ima šansu, ali bolju šansu dobijaju roditelji sa boljom vrednošću fitnes funkcije. Prvo se na osnovu vrednosti fitnes funkcije poredaju u rang (od 1 koji je najbolji do *number_of_agents* koji je sa najmanjom fitnes vrednošću), te se svaki od njih množi sa nasumičnim brojem iz uniformne distribucije od 0 do 1. Za kraj se nad ovim brojevima vrši softmax [10]. Onda se na osnovu šanse tih brojeva biraju roditelji (moraju biti različita).

Za mutaciju genotipa dece, na osnovu *mutation_chance* se odlučuje da li će se detetovi genotipi mutirati ili ne (predstavlja broj između 0 i 1). Dete se mutira tako što se uzme nasumičan broj iz uniformne distribucije između 0 i 1, koji ima za cilj da promeni vrednosti kandidatskih parametara u rasponu -10% do +10% u zavisnosti od nasumičnog broja.

Za ukrštanje genotipa roditelja, ideja je takva da se nađe nasumičan broj iz uniformne distribucije između 0 i 1 i primene sledeće formule za ukrštanje dva deteta koje dva roditelja stvaraju za genotipe dece

$$newParam1 = param1 * c + param2 * (1 - c)$$
$$newParam2 = param2 * c + param1 * (1 - c)$$

Za elitne kandidate, biraju se direktno oni koji imaju najveći fitnes rezultat, broj elitnih kandidata definisan je prema *top_limit_agents* parametru. Elitni kandidati prelaze u novu generaciju netaknuti, odnosno ne prolaze kroz proces mutiranja.
Za kreiranje nove generacije se koriste tehnike kreiranja novih kandidata standardnim putem ukrštanja i mutiranja preko roditelja, a ostatak se popuni elitnim kandidatima, tako da veličina populacije iz generacije u generaciju ostane ista.

3. EKSPERIMENTI I REZULTATI

Za adekvatno treniranje algoritama dubokog učenja sa potkrepljenjem potrebni su i adekvatni hiperparametri koji će algoritam da dovedu do zadovoljavajućeg rešenja.
Za treniranje *TRPO* algoritma [1], korišteni su hiperparametri dati u tabeli ispod. Naime, ovi parametri birani su eksperimentisanjem i isprobavanjem, kao i upotrebom preporučenih vrednosti koji za većinu problema radi zadovoljavajuće.

TRPO hiperparametri	
Ime parametra	Vrednost parametra
learning_rate	$2.5*10^{-4}$
delta	$[3*10^{-1}, 3*10^{-2}, 3*10^{-3}, 3*10^{-4}]$
cg_delta	$1*10^{-2}$
cg_iterations	10
alpha	0.99
backtrack_steps_num	100
critic_epoch_num	20
num_of_timesteps	4800
max_timesteps_per_episode	1600
gamma	0.99
num_of_epochs	40900
mean_episode_reward	~526
time_for_training (days)	10

Prilikom treniranja *TRPO* algoritma, pojavljuju se dva glavna problema:

- potreba da se $delta$ hiperparametar menja u toku treniranja, jer na početku treniranja odgovara veća vrednost (ako zadržimo manju, učenje ide sporo do nivoa da se i ne trenira), dok posle nekog vremena zahteva manju vrednost jer se zahteva manji region poverenja radi optimalnijeg treniranja (ako zadržimo veliku vrednost, očekivani povrat epizoda po epohama ili stagnira ili opada)
- dugotrajan proces obučavanja, koji ukazuje na to da hiperparametri mogu bolje biti optimizovani tako da je proces manji

Grafici treniranja i prosečnog povrata po epohama TRPO algoritma dati su na [7].
Za treniranje *PPO* algoritma [3], korišteni su hiperparametri dati u tabeli ispod. Naime, ovi parametri birani su eksperimentisanjem i isprobavanjem, kao i upotrebom preporučenih vrednosti koji za većinu problema radi solidno.

PPO hiperparametri	
Ime parametra	Vrednost parametra
learning_rate	$2.5*10^{-4}$
num_of_timesteps	4800
max_timesteps_per_episode	1600
critic_epoch_num	20
gamma	0.99
clip_range	0.2
num_of_epochs	9000
mean_episode_reward	~558
time_for_training (days)	2

PPO kao algoritam za dati problem se umnogome pokazao kao bolji izbor. U slučaju ovog algoritma, nema potrebe za manuelnim menjanjem parametara, niti za dugim obučavanjem. Grafici treniranja i prosečnog povrata po epohama PPO algoritma dati su na [7].
S obzirom na napomenute probleme koji se javljaju kod *TRPO* algoritma, cilj je sada inkorporirati pristup genetskog algoritma tako da potencijalno reši nedostatke koji se javljaju prilikom obučavanja *TRPO* algoritma na ovom problemu. Cilj genetskog biće da pronađe optimalne hiperparametre koji će dovesti do efikasnijeg treniranja i rešiti navedene probleme.
Kod *TRPO* algoritma, jedini hiperparametar koji zahteva da se manuelno menja tokom treniranja jeste $delta$, i to na opadajući način. Ideja je predstaviti $delta$ kao kvadratnu funkciju koja opada u zavisnosti od trenutnog broja epohe na način:

$$delta = \frac{1}{deltaa2 * n^2 + deltaa1 * n + deltaa0}$$

gde je n broj trenutne epohe, a parametri $deltaa2, deltaa1, deltaa0$ ciljani parametri za optimizaciju nad genetskim algoritmom.

Za treniranje genetskog algoritma, korišteni su hiperparametri dati u tabeli ispod. Naime, ovi parametri birani su eksperimentisanjem i isprobavanjem, kao i upotrebom preporučenih vrednosti koji za većinu problema radi solidno. Konkretan parametar $number\ of\ iterations$ je odabran na datu vrednost kao najbolji odnos vremenske zahtevnosti obučavanja jednog kandidata i broja iteracija (epoha) koliko će jedan kandidat da se trenira.

GA hiperparametri	
Ime parametra	Vrednost parametra
number_of_agents	15
number_of_iterations	70
top_limit_agents	1
mutation_chance	0.1
number_of_generations	17
delta_a2	0.01487870062184785
delta_a1	0.0005061652202640241
delta_a0	8.163872665158063
time_for_training (days)	4.5

Rezultati koji su postignuti na optimizovanim hiperparametrima nisu bolji, čak su i znatno lošiji sa daljim epohama od onih koji se dobijaju sa menjanjem manuelno hiperparametra *delta*, a glavni razlog tome je što vrednost $delta$ postaje premali posle nekog broj epohe (zato što je funkcija predstavljena pomoću nje), te se treniranje izvršava sporo, skoro pa nikako, što ne doprinosi rešenju efikasnosti i optimalnosti, dok je rešenje da se ne menjaju parametri tokom treniranja zadovoljeno. Ovakvim pristupom rešenje nikada ne bi konvergiralo ka krajnjem rešenju jer je vrednost $delta$ premalo i sa sve većim brojem epoha bi težilo nuli odnosno nepostojećem regionu poverenja, tako da učenja nema.
Kao rešenje koje će dovesti do konvergencije, uzimajući u obzir prvobitni eksperiment nad *TRPO* algoritmom jeste promena funkcije koja opisuje deltu na sledeći način:

$$delta = max(\frac{1}{deltaa2*n^2 + deltaa1*n + deltaa0}, deltamin)$$

gde je sada $deltamin$ jednak minimalnoj pozitivnoj vrednosti kojoj može parametar $delta$ da bude jednak, u cilju prevencije premalih vrednosti.
Gledajući novi predlog $delta$ funkcije, kao $deltamin$ se može odabrati najmanja vrednost iz prvog eksperimenta nad *TRPO* algoritmom, a za preostale vrednosti koristeći preostale hiperparametre dobijene genetskim algoritmom. Takav *TRPO* algoritam uspešno konvergira ka dobrim rezultatima i rešava problem potrebe za menjanjem parametara u toku treniranja. Dužina trajanja treniranja je i dalje duga (9 dana), što je u poređenju sa *PPO* algoritmom neefikasno rešenje. Grafici treniranja i prosečnog povrata po epohama optimizovanog TRPO algoritma dati su na [8].

4. ZAKLJUČAK

Kao što se može videti u radu, algoritmi dubokog učenja sa potkrepljenjem mogu biti efektivni za rešavanje problema učenja robota da hoda, te oba primenjena uspešno rešavaju postavljeni problem.
Kao ideja za rešavanja efikasnosti *TRPO* algoritma uz pomoć genetskog algoritma, definitivno bi potreba bila koristiti više iteracija po svakom kandidatu (reda par stotina pa možda čak i do hiljada), jer tek tada bi genetski mogao da pronađe rešenje koje će da efektivno trenira i na početnim epohama i posle nekoliko hiljada epoha i time se svrsta u rang efikasnosti kao što je to postigao *PPO*. Ovo bi zahtevalo velike vremenske i računarske resurse da bi se izvršilo uspešno do kraja. Svakako, *PPO* kao algoritam je generalno bolji u odnosu na *TRPO*, što je takođe i pokazano kroz ovaj rad koliko je efikasniji u treniranju.

5. LITERATURA

[1] Schulman, John, et al. "Trust region policy optimization." *International conference on machine learning*. PMLR, 2015.
[2] Fletcher, Roger. "Conjugate gradient methods for indefinite systems." *Numerical analysis*. Springer, Berlin, Heidelberg, 1976. 73-89.
[3] Schulman, John, et al. "Proximal policy optimization algorithms." *arXiv preprint arXiv:1707.06347* (2017).
[4] https://gym.openai.com/
[5] https://pybullet.org/wordpress/
[6] https://pytorch.org/
[7] https://github.com/reinai/HumanoidRobotWalk
[8]https://github.com/sovaso/GeneticAlgorithmForHumanoidRobotWalk
[9] Reynolds, Douglas A. "Gaussian mixture models." *Encyclopedia of biometrics* 741 (2009): 659-663.
[10] Gao, Bolin, and Lacra Pavel. "On the properties of the softmax function with application in game theory and reinforcement learning." *arXiv preprint arXiv:1704.00805* (2017).

Kratka biografija:

Vasilije Pantić rođen je u Novom Sadu 1997. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Softversko inženjerstvo odbranio je 2021.god.
kontakt: vpantic10@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.313
DOI: https://doi.org/10.24867/15BE29Stojanovic

MODELI TRANSFORMATORA I ASINHRONIH MAŠINA UZ UVAŽAVANJE VIŠIH HARMONIKA

MODELING OF TRANSFORMERS AND INDUCTION MACHINES WITH REGARD OF HIGHER HARMONICS

Marko Stojanović, Marko Vekić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ENERGETIKA

Kratak sadržaj – *Ovaj rad bavi se modelovanjem odziva asinhronih mašina i transformatora u uslovima izobličenog mrežnog napona. Razmatrane su veličine kao što su gubici i električni moment.*

Ključne reči: *transformatori, asinhroni motori, viši harmonici*

Abstract – *This paper deals with the operation of Induction motors and transformers under distorted grid conditions.*

Keywords: *Induction motor, transformer, higher harmonics.*

1. UVOD - HARMONICI I KVALITET ELEKTRIČNE ENERGIJE

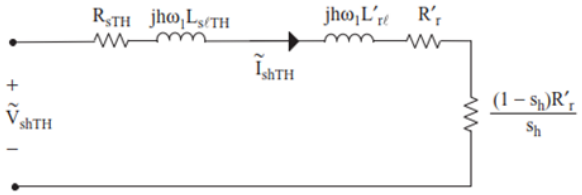
Nelinearni potrošači, odnosno transformatori, rotirajuće obrtne električne mašine, FACTS uređaji, i komponente energetske elektronike su neki od uzroka pojave nesinusnih talasnih oblika struje i napona, što uzrokuje loš kvalitet električne energije.

Glavne posledice viših harmonika su neispravan rad upravljačkih uređaja, telefonske smetnje, dodatni gubici na vodovima (pri osnovnom i višim harmonicima), smanjen životni vek i uvećani gubici uređaja (npr. transformatori, obrtne mašine, i kondenzatorske baterije), i neispravan rad korisničkih uređaja.

Loš kvalitet električne energije ima mnoge štetne posledice na uređaje sistema i krajnje potrošače. Ono što čini ovu pojavu dodatno lošom je to što njeni efekti ostaju neopaženi sve dok ne dođe do kvara. Postoje mnogi standardi i srodna dokumenta koja se bave problemima kvaliteta električne energije. Lista dostupnih dokumenata o problemima kvaliteta omogućava lakšu potragu za potrebnim informacijama.

2. MODEL ASINHRONE MAŠINE U SLUČAJU IZOBLIČENJA NAPONA NAPAJANJA

Za struju i obrtni moment h-tog harmonika, na osnovu aproksimacija i Tevenenove predstave (slika 2.1), dobija se:



Slika 2.1 *Tevenenovo ekvivalentno električno kolo asinhrone mašine za h-ti harmonik*

$$\tilde{I}_{shTH} = \frac{\tilde{V}_{shTH}}{\left(R_{shTH} + \frac{R'_r}{s_h}\right) + jhw_1(L_{slTH} + L'_{rl})} \#(2.1)$$

$$T_{eh} = \frac{1}{w_{sh}} \frac{q_1 V_{shTH}^2 \frac{R'_r}{s_h}}{\left(R_{sTH} + \frac{R'_r}{s_h}\right)^2 + (hw_1)^2(L_{slTH} + L'_{rl})^2} \#(2.2)$$

slično, za obrtni moment usled osnovnog harmonika dobija se:

$$T_{e1} = \frac{1}{w_{s1}} \frac{q_1 V_{s1TH}^2 \frac{R'_r}{s_1}}{\left(R_{sTH} + \frac{R'_r}{s_1}\right)^2 + (w_1)^2(L_{slTH} + L'_{rl})^2} \#(2.3)$$

Klizanje asinhrone mašine pri prvom, kao i pri višim harmonicima, dato je sledećim relacijama:

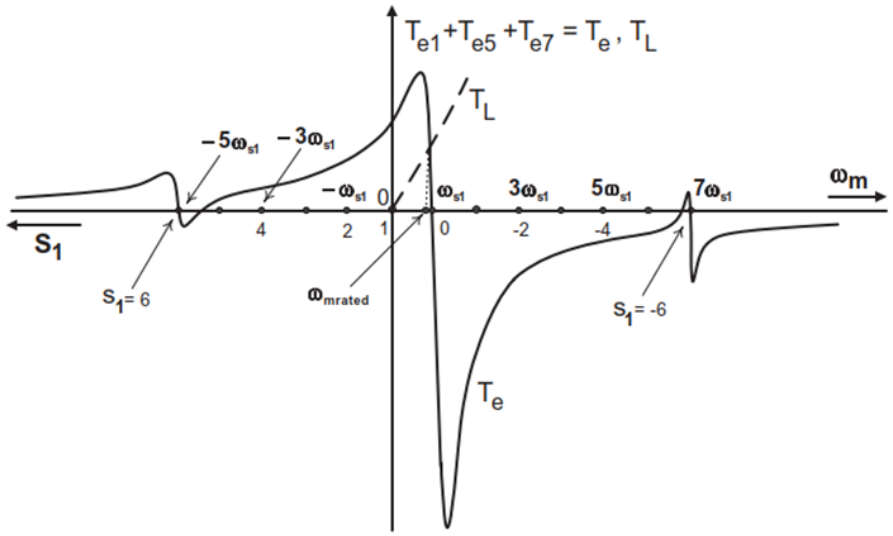
$$s_1 = \frac{w_{s1} - w_m}{w_{s1}} \#(2.4)$$

$$s_h = \frac{hw_{s1} - w_m}{hw_{s1}} \#(2.5)$$

gde $w_{s1} = \frac{w_1}{p/2}$ predstavlja mehaničku sinhronu ugaonu brzinu osnovnog harmonika, a w_m je mehanička ugaona brzina rotora.

Da bismo usvojili smer rotacije harmonijskih magnetopobudnih sila, u nastavku pretpostavljamo da prvi rotira u smeru kazaljke na satu (negativan matematički smer), peti obrnuto od smeta kazaljke na satu (pozitivan matematički smer), i sedmi, kao i prvi, u negativnom smeru.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Marko Vekić, vanr. prof.



Slika 2.2 *Obrtni moment asinhrone mašine za osnovni, peti i sedmi harmonik u funkciji ugaone brzine i klizanja osnovnog harmonika* s_1

Superpozicija osnovnog $(h = 1)$, petog $(h = 5)$, i sedmog harmonika $(h = 7)$ obrtnog momenta $T_e = T_{e1} + T_{e5} + T_{e7}$ prikazana je na slici 2.2. U tački $w_m = w_{mrated}$ ukupni električni obrtni moment T_e je jednak obrtnom momentu opterećenja T_L, odnosno $T_{e1} + T_{e5} + T_{e7} = T_L$, gde T_{e1} i T_{e7} predstavljaju motorske obrtne momente, a T_{e5} kočioni obrtni moment.

3. SMANJIVANJE SNAGE TRANSFORMATORA USLED POJAVE VIŠIH HARMONIKA

Poznato je da mora doći do smanjenja snage transformatora kada se vrši napajanje nelinearnih opterećenja. Jedna od metoda koja omogućava opterećenje višim harmonicima je projektovanje transformatora s tzv. K-sačiniocem (faktorom), koji predstavlja meru uticaja viših harmonika u struji opterećenja na transformator.

Gubici opterećenja transformatora u uslovima viših harmonika su:

$$P_{LL} = \sum_{h=1}^{h\,max} I_h^2 + \left(\sum_{h=1}^{h\,max} I_h^2\, h^2\right) P_{EC-R} \#(3.1)$$

Prvi i drugi sabirak sa desne strane iz prethodne jednačine predstavljaju I^2R, i gubitke usled vrtložnih struja, redom. Ukoliko usvojimo da je $P_{LL} = P_{LL-R}$ važi:

$$1 + P_{EC-R} = \sum_{h=1}^{h\,max} I_h^2 + \left(\sum_{h=1}^{h\,max} I_h^2\, h^2\right) P_{EC-R} \#(3.2)$$

ukoliko uvedemo K-sačinilac kao:

$$K = \frac{\sum_{h=1}^{h\,max} I_h^2\, h^2}{I_R^2} \#(3.3)$$

onda važi:

$$1 + P_{EC-R} = \sum_{h=1}^{h\,max} I_h^2 + K\left(\frac{\sum_{h=1}^{h\,max} I_h^2}{\sum_{h=1}^{h\,max} I_h^2}\right) P_{EC-R} \qquad (3.4)$$

Rešavajući po: $\sum_h^{h\,max} I_h^2$,

$$\sum_{h=1}^{h\,max} I_h^2 = \frac{1 + P_{EC-R}}{1 + K\frac{I_R^2}{\sum_{h=1}^{h\,max} I_h^2}(P_{EC-R})} \qquad (3.5)$$

Dakle, kvadratni koren maksimalne vrednosti struje opterećenja koju transformator može da isporuči je:

$$I_{max}^j = \sqrt{\frac{1 + P_{EC-R}}{1 + K\frac{I_R^2}{\sum_{h=1}^{h\,max} I_h^2}(P_{EC-R})}} \qquad (3.6)$$

Koristeći K faktor i informacije o parametrima transformatora, kvadratni koren maksimalne dozvoljene vrednosti struje transformatora može se definisati i na sledeći način:

$$I_{max}^j = \sqrt{\frac{R_{DC} + (R_{EC-R})(1-K) - \frac{(\Delta P_{fe} + \Delta P_{OSL})}{I_R^2}}{R_{DC}}} \qquad (3.7)$$

gde: $R_{DC} = R_{DCprimarno} + R'_{DCsekundarno}$ je ukupni DC otpor namotaja transformatora. R_{EC-R} je pretpostavljeni dodatni otpor usled vrtložnih struja. Dalje:

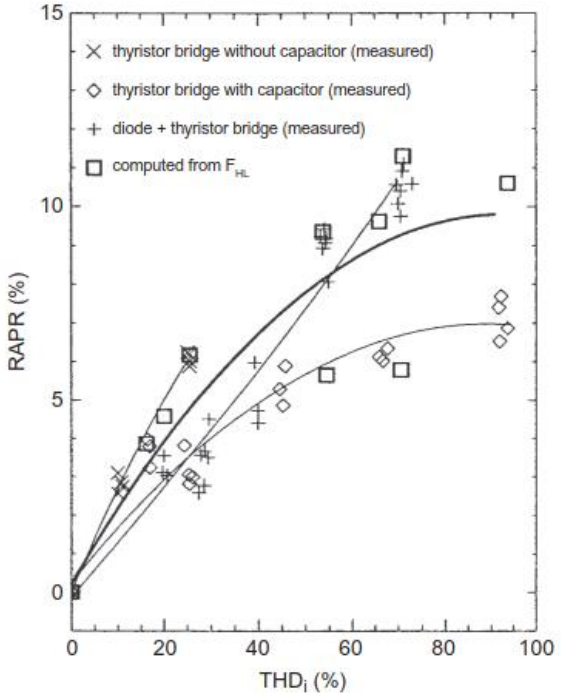
$$\Delta P_{fe} = \sum_{h=1}^{h\,max} P_{feh} - P_{feR}, \qquad (3.8)$$
$$\Delta P_{OSL} = \sum_{h=1}^{h\,max} P_{OSLh} - P_{OSLR}.$$

ΔP_{fe} je razlika između ukupnih gubataka u gvožđu (uključujući harmonike) i procenjenih gubitaka u gvožđu bez harmonika. ΔP_{OSL} je razlika između ukupnih preostalih gubitaka (uključujući harmonike) i procenjenih ostalih gubitaka bez harmonika.

Smanjenje prividne snage (RAPR) je:

$$RAPR = 1 - \left(V_{2rms}^{nelinerno}/V_{2rms}^{rat}\right) I_{max}^j \qquad (3.9)$$

gde su $V_{2rms}^{nelinerno}$ i V_{2rms}^{rat} ukupna vrednost kvadratnog korena napona sekundara transformatora uključujući harmonike, i pretpostavljena vrednost kvadratnog korena sekundarnog namotaja bez harmonika, redom.



Slika 3.1 *Izmereno smanjenje opsega prividne snage (RAPR) jednofaznog transformatora snage 25kVA u funkciji ukupnog harmonijskog izobličenja struje (THD_i), gde su treći i peti harmonik izraziti.*

4. ZAKLJUČAK

Cilj ovog rada bio je izvršiti pregled i analizu modela transformatora i asinhronih mašina u harmonijskom okruženju. Izbor je pao na ova dva uređaja jer su transformatori izuzetno bitan element sistema, a asinhroni motor elektromotornih pogona, i kao takvi u značajnoj meri utiču na kvalitet.

U radu su dati analitički izrazi i proračuni dejstva navedenih uređaja u uslovima poremećaja koji dolaze iz električne mreže, ali dat je i osvrt na njihovu osobinu da unose više harmonike, pre svega usled nelinearnih pojava u njihovom magnetnom jezgru.

5. LITERATURA

[1] Stoffel, T.; Andreas, A. (2008). Sun Spot One (SS1): San Luis Valley, Colorado (Data). NREL Report No. DA-5500-56507. http://dx.doi.org/10.5439/1052452 Accessed May 8, 2015.This image has been reprinted with permission from the National Renewable Energy Laboratory.

[2] Allen E, Kosterev DN, Pourbeik P. Validation of power system models. In: 2010 IEEE Power and Energy Society general meeting, Minneapolis, MN. doi:10.1109/PES.2010.5589874.

[3] Miller NW, Sanchez Gasca JJ, Price WP, Delmerico RW. Dynamic modeling of GE 1.5 and 3.6 MW wind turbine-generators for stability simulations, 0-7803-7989-6/03@2003 IEEE.

[4] Wang Q, Chang L. An intelligent maximum power extraction algorithm for inverter-based variable speed wind turbine systems. IEEE Trans Power Electron 2004;19(5):1242–9.

Kratka biografija:

Marko Stojanović je rođen je u Novom Sadu 1994. god. Završio gimnaziju Isidora Sekulić u Novom Sadu. Osnovne akademske studije i odbranu diplomskog rada završio na Fakultetu tehničkih nauka 2018. godine. Trenutno zaposlen u kompaniji Strabag.

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

Зборник радова Факултета техничких наука, Нови Сад

UDK: 004.4
DOI: https://doi.org/10.24867/15BE30Vasiljevic

FILE TRANSFER ФУНКЦИОНАЛНОСТ DNP3 ПРОТОКОЛА И ЊЕГОВА ПРИМЕНА У *DSCADA* СОФТВЕРСКОМ СИСТЕМУ

FILE TRANSFER FEATURE OF DNP3 PROTOCOL AND ITS APPLICATION IN *dSCADA SOFTWARE SYSTEM*

Немања Васиљевић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *У овом раду описан је DNP3 протокол и његова конкретна имплементација у dSCADA софтверском систему са нагласком на пренос датотека (File transfer), функционалност коју, као опцију, имају само протоколи новије генерације. У самом стандарду, конкретна примена File transfer-а није специфично описана и дефинисана, што омогућава разноврсну примену ове функционалности у реалном раду. Једна од могућих примена је описана у овом раду, која омогућује упис специфичних догађаја који су се десили у процесном систему и аквизицију истих путем File transfer-а, чиме се омогућује накнадна детаљна анализа тих догађаја.*

Кључне речи: *SCADA, DNP3 протокол, File transfer*

Abstract – *This paper describes DNP3 protocol and their concrete usage in dSCADA software system with accent on File transfer functionality. File transfer is advance functionality supported only by newer generation protocols. Concrete usage of File transfer functionality is not specifically described and defined, which enables various usage of this functionality in real work. One of the usages is described in this paper, and it enables the recording of specific events occurring in the system being monitored, later File-transfer acquisition and detailed post-mortem analysis of recorded events.*

Keywords: *SCADA, DNP3 protocol, File transfer*

1. УВОД

Технолошки напредак у области микроелектронике и информационих технологија, омогућио је убрзани развој система за управљање индустријским процесима и постројењима уз помоћ рачунара. Тако су аутоматизована производна постројења почела да се развијају у раној фази развоја рачунарства, средином шездесетих година, и представљала су електромеханичке системе са рачунарским управљачким језгром. Овакви системи, специфичне намене и структуре у литератури се најчешће означавају термином SCADA (*Suervisory Control And Data Acquisition*) [2].

SCADA системи постали су изузетно важни за већину индустрија широм света јер контролишу критичне инфраструктуре [1] као што су електричне мреже, водоводи, гасоводи…

File transfer је једна од функционалности подржаних у DNP протоколу која омогућава пренос фајлова између SCADA сервера (*master*) и процесног контролера (*slave*).

2. *SCADA*

SCADA (Supervisory Control And Data Acquisition) подразумева надзор, контролу и прикупљање података.

SCADA систем је аквизиционо-управљачки систем. Аквизиција података подразумева добављање вредности са дигиталних и аналогних мерних уређаја из поља. Аквизиција је неизоставан процес у свим SCADA системима јер се све одлуке доносе на основу прикупљених и обрађених података [2].

Појавом дигиталних рачунара створила се могућност да се комплексност система далеко лакше савлада, развојем програма који ће прикупљати мерне податке, моделовати реално стање индустријског постројења и обезбедити квалитетнији надзор и управљање системом.

Традиционални системи су се састојали од једног *PLC-a* који је контролисао индустријско постројење [2]. Временом се јавила потреба за управљањем системима са дистрибуираним ресурсима па је SCADA прерасла у дистрибуирани систем. Данас су SCADA системи присутни у скоро свим критичним инфраструктурама [1].

3. ИНДУСТРИЈСКИ ПРОТОКОЛИ

Индустријски протоколи представљају групу комуникационих протокола за *SCADA* системе. Традиционално индустријски протоколи су често везани за специфичну област примене, још чешће за произвођача опреме [2].

У Америци је доминантан *DNP3* протокол, док су у Европи *IEC 60870-5-101* и *IEC 6087-5-104* [3]. У старијим системима још увек се користе *Modbus, Fieldbus,* као и многи други власнички протоколи [3].

4. DNP3

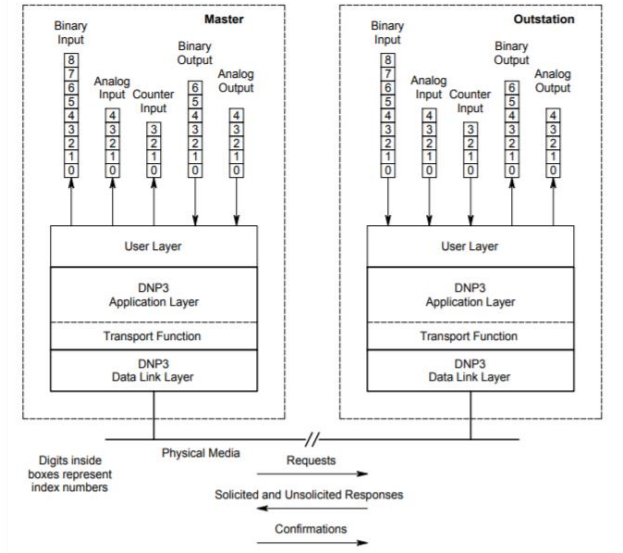
DNP3 (*Distribution Network Protocol*) представља сет комуникационих протокола који се користе за размену информација између компоненти у системима за аутоматизацију процеса. Његова главна

НАПОМЕНА:
овај рад проистекао је из мастер рада чији ментор је био проф. др Драган Кукољ.

улога је у употреби у компанијама које поседују системе као што су системи за дистрибуцију електричне енергије и водоснабдевања. Развијен је за комуникацију између различитих врста опреме за прикупљање и контролу података. Овај протокол игра главну улогу у SCADA системима, где се користи за комуникацију између MASTER станица и RTU-ова и интелигентних електронских уређаја (IDEs).

DNP3 протокол је иницијално настао док су интернет протоколи и сам ISO-OSI скуп протокола био у развоју, када није постојала адекватна мрежна инфраструктура и сам мрежни слој. Зато је изворни DNP3 протокол користио редуковани ISO-OSI модел, такозвани EPA (*Enhanced Performance Architecture*) модел са циљем да се одржи висок степен сигурности у преносу података.

Овај модел се састоји од 3 слоја: апликационог слоја, слоја канала и физичког слоја. Овакав модел комуникације код DNP3 протокола приказан је на слици 4.1.

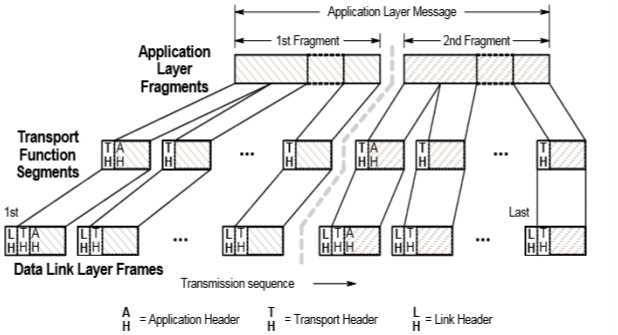


Слика 4.1 Master – Outstation модел комуникације DNP3 протокола

Сваки слој обезбеђује скуп функција које обезбеђују комуникацију са слојем на истом нивоу уређаја са којим се комуницира, при томе ослањајући се на нижи слој у обављању више примитивних функција [4].

4.1 Сегментација поруке

Слика 4.2 приказује фрагментовану поруку апликационог слоја по транспортној функцији, као и енкапсулацију података од апликационог слоја до слоја података. Такође приказује релативне положаје заглавља апликационог слоја, заглавља транспортне функције као и заглавља слоја. Сваки од горе наведених слојева енкапсулира податке добијене од слоја који се налази изнад њега и на добијену поруку лепи заглавље које носи информације које омогућавају правилну обраду поруке на слоју истог нивоа на пријемној страни.

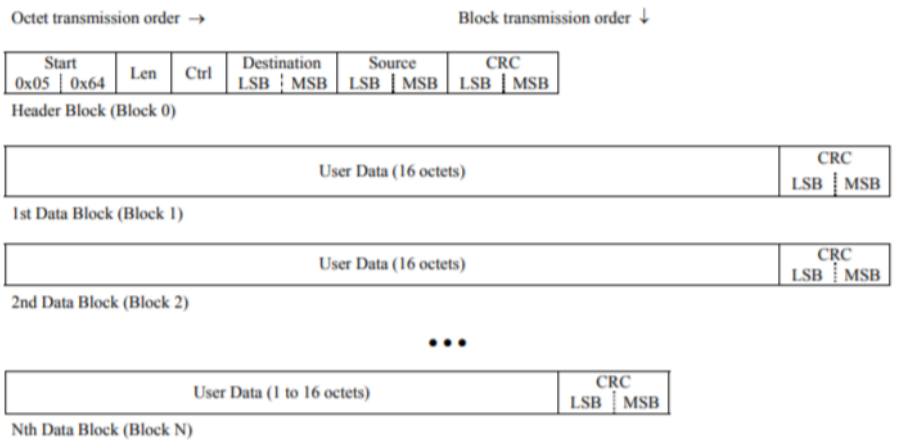


Слика 4.2 *Поступак сегментирања поруке*

4.2 Сегмент слоја података везе (Data Link Layer Frame)

Слој података везе пружа интерфејс између подслоја транспортне функције и апликационог слоја и физичког медија за пренос података или мрежног слоја. Главни допринос фрагмента слоја података је могућност адресирања удаљених станица и могућност откривања грешке. Протокол је дизајниран да делује и у бајтовском режиму рада и у пакетско орјентисаним мрежама где је основна јединица преноса података пакет, као што су *TCP/IP, UDP/IP*.

Оквир везе података има заглавље фиксне дужине, иза кога следе опциони блокови података. Блокови података су дужине 16 бајтова иза сваког блока следи 16-битно поље које садржи CRC код за откривање грешке у преносу података (Слика 4.3).

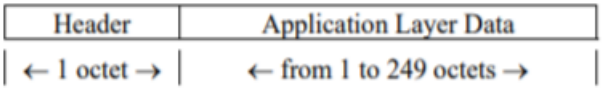


Слика 4.3 *DNP3 формат оквира података*

4.3 Сегмент транспортне функције (Transport Function Segment)

Величина DNP3 фрагмента поруке апликационог нивоа може бити већа од дозвољеног броја октета дозвољеног у једном оквиру слоја везе. Стога сврха транспортне функције је да фрагмент апликационог слоја растави на јединице података (Transport Segments) које су погодне за пренос са предајне стране и на исти начин на пријемној страни састави све пристигле фрагменте у оригиналну поруку.

На месту предаје информације, подаци са апликационог нивоа су растављени на мање делове којима се додаје заглавље транспортне функције (Transport header). Заглавље транспортне функције и подаци са апликационог слоја заједно чине фрагмент података транспортне функције (Слика 4.4), који се прослеђује слоју података.

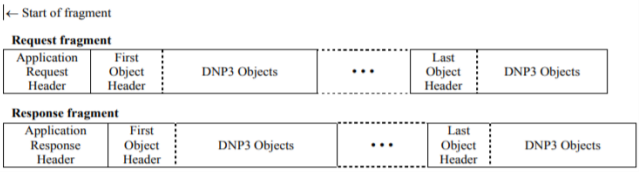


Слика 4.4 Фрагмент података транспортне функције

4.4 Сегмент апликационог слоја (Application Layer Fragment)

Фрагмент је блок узастопних осмобитних вредности (у даљем тексту октета), који садрже информације захтева или одговора који се размењују између мастер стране и удаљене станице.

Сваки фрагмент садржи функцијски код који одређује како ће прималац обрадити пристигли фрагмент, нула или више заглавља података, објекте испуњене подацима као и информације да ли су примљени фрагменти пристигли у одговарајућем редоследу (Слика 4.5). Максимална дужина фрагмента одређује да ли ће порука која се шаље бити послата као један или више фрагмената. Мање поруке могу бити послате као један фрагмент.



Слика 4.5 *Структура фрагмента захтева и одговора*

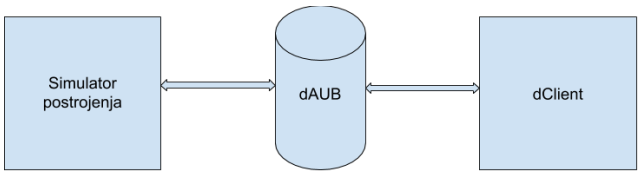
5. dSCADA СОФТВЕРСКИ СИСТЕМ

dSCADA софтвер је програмски пакет школске SCADA-е, који чине три основне компоненте приказане на слици (Слика 5.1):

Симулатор процесног контролера, за потребе развоја DNP3 протокола коришћен је DNP3 симулатор објашњен у претходном поглављу,

Аквизиционо управљачка станица – dAUB,

Клијентска радна станица са графичком спрегом ка оператеру – dClient [2].



Слика 5.1 Компоненте dSCADA апликације

5.1 Клијентска радна станица (dClient)

Клијентска радна станица dClient (оператерска конзола) је апликација која обезбеђује табеларни и графички приказ тренутног стања SCADA система, визуелизацијом оних атрибута процесних величина које су оператеру најинтересантнији.

То значи да многи елементи dSCADA модела нису доступни преко оператерске спреге, јер нису претерано важни у лабораторијској примени или су доступни на неки други начин (кроз развојно окружење).

Поред табеларног приказа, dClient апликација садржи и графички приказ процесног система. Графички приказ се састоји од технолошких дијаграма (шема) процесног система на којима су приказани технолошки елементи постројења. Елементи система повезани са SCADA системом се динамички освежавају, што значи да се њихово стање мења у виду промене боје, додавање текста и променом графичког облика. Остатак дијаграма као технолошка подлога графичког приказа постројења, је непроменљив по иницијалном исцртавању на екрану [2].

5.2 Аквизиционо управљачки блок (dAUB)

Аквизиционо управљачки блок dAUB је централна програмска компонента dScada система која врши функцију SCADA сервера у реалном времену. Иза dAUB апликације стоји слојевита програмска подршка, која прати основне функционалне целине SCADA аквизиционо управљачке станице. Централни део dAUB софтвера представља меморијска база података у реалном времену, која чува променљиве dSCADA модела процеса и све остале структуре података неопходне за рад [2].

5.3 Симулатор постројења (simPLC)

RTU симулатор је програмска компонента, део dSCADA софтверског система чији је задатак да замени физички процесни контролер уз обезбеђење идентичне комуникационе спреге [2].

Суштински задатак SCADA симулатора јесте динамичка симулација окружења SCADA система, тачније процесног окружења које одређује понашање SCADA надзорно управљачке станице [2].

Симулација RTU уређаја има два важна аспекта: симулацију комуникације и симулацију понашања [2].

Симулација комуникације је релативно једнозначан, што не значи и једноставан задатак. Потребно је реализовати RTU станицу неког индустријског протокола, инверзну у односу на протокол који користи SCADA станица.

Стандардно RTU симулатор прима упите SCADA станице, анализира их и шаље адекватне одговоре [2].

Симулација понашања RTU уређаја је, у ствари, симулација постројења иза њега. Промене у постројењу настају као последица одвијања технолошког процеса, или као реакција на управљачке команде SCADA система [2].

6. FILE TRANSFER ФУНКЦИОНАЛНОСТ

File transfer функционалност подразумева пренос комплетних датотека, односно читање и писање датотека између *master* и *slave* станице од почетка датотеке па све до краја.

File transfer увек иницира *master* што подразумева могућност читања и писања фајлова само на удаљеној станици, али не и обратно. У dSCADA софтверском систему имплементирана је функционалност читања фајлова са удаљене станице.

Читање датотека путем DNP3 протокола састоји се од следећих трансакција:

- Опционална аутентификација
- Отварање фајла који читамо
- Један или више захтева за читање фајла
- Затварање фајла

Ако систем захтева кључ за потзврду идентитета за отварање датотеке, мастзер издаје захтев за потврду идентитета.

Отварање датотеке, читање и затварање захтева враћање објекта догађаја од удаљене станице. Ако удаљена станица може одмах да одговори, она враћа одговарајући објекат, у супротнм ако удаљена станица не може одмах да пошаље одговор, она одговара са поруком која не садржи објекте података.

6.1 *Save event* функционалност

Конкретна примена File transfer функционалности није специфично описана и дефинисана, што омогућава разноврсну примену ове функционалности у реалном раду.

Једна од примена која је имплементирана у dSCADA софтверски систем јесте чување специфичних догађаја који су од интереса за перзистенцију у фајл на удаљеној стани. То омогућава прикупљање тих фајлова са свих удаљених станица које надзиремо и њихову детаљну реконструкцију анализу.

Од конкретне имплементације симулатора удаљене станице односно од конфигурације стварног RTU уређаја зависи који догађаји су од интереса да се перзистирају у фајл.

У случају dSCADA система кроз конфигурациони параметар (SAVE_EVENT 1) је могуће омогућити или онемогућити креирање фајла за упис догађаја, као и упис догађаја у исти.

7. ЗАКЉУЧАК

Основна замисао овог рада била је имплементација напредних функционалности DNP3 протокола, међу којима је *File transfer* и њена реална примена. Функционалност је имплементирана и на *master* станици као и на симулатору.

За реализацију рада било је неопходно детаљно упознавање свих функционалности DNP3 протокола, њихову имплементацију, као и упознавање архитектуре dSCADA софтверског система.

За саму имплементацију протокола коришћени су алати *Outstation DNP Simulator* и *Cliet DNP Simulator* као референтни алати за развој, како би поруке које се размењују биле у потпуности у складу са захтевима протокола.

Наставак и унапређење dSCADA софтверског система била би развој алата који би олакшао анализу фајлова са удаљених станица односно анализу забележених догађаја.

8. ЛИТЕРАТУРА

[1] Kyle Coffey, Richard Smith, Leandros Maglaras and Helge Janicke, *Vulnerability Analysis of Network Scanning on SCADA Systems*, 2018

[2] Branislav Atlagić, *Softver sa kritičnim odzivom, projektovanje SCADA sistema*, 2015

[3] Frances Cleveland, *IEC TC57 Security Standards for the Power System's Information Infrastructure – Beyond Simple Encription*, 2006

[4] *Никола Живковић – Једно решење имплементације ДНП3 протокола, 2011*

[5] *DNP_SpecificationDocuments_20100223*

[6] *G.Clarke, D.Reynders, E.Wright, Practical Modern SCADA Protocols: DNP3, 60870.5 and Related Systems, Elsevier 2004.*

[7] *S.Boyer, SCADA: Supervisory Control and Data Acquisition, ISA, 2009.*

[8] *D.Bailey, E.Wright, Practical SCADA for Industry, Elsevier, Burlington 2003.*

[9] *J.F.Kurose, K.W.Ross, Computer Networking: A Top-Down Approach, Pearson Education, Boston 2010.*

Кратка биографија:

Немања Васиљевић рођен је 1996. године у Ужицу. Завршио је Средњу Техничку школу у Ужицу 2014. године. Исте године уписао је Факултет Техничких наука у Новом Саду. Дипломски рад под називом *Имплементација DNP3 протокола у dSCADA софтвер* одбранио је 2019 године.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.317
DOI: https://doi.org/10.24867/15BE31Sas

IOT MERNO-INFORMACIONI SISTEM ZA PAMETNI PLASTENIK

IOT MEASUREMENT AND INFORMATION SYSTEM FOR SMART GREENHOUSE

Milan Šaš, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu prikazan je realizovani merno-informacioni sistem čija primena je u objektima poput staklenika ili plastenika a za cilj ima merenje zadatih fizičkih veličina, ostvari bežični prenos, obradu i prikaz podataka kao i mogućnost automatizacije rada uređaja koji se nalaze u objektu, poput pumpi za vodu, ventila na crevima, grejača, i sličnih uređaja koji mogu da se nađu u objektu. Sistem je projektovan kao „low cost" rešenje i koristi široko dostupne uređaje i senzore.*

Ključne reči: *SHT35, SHT31, TEMT6000, NRF24L01, Arduino Nano, Raspberry Pi, Python*

Abstract – *This paper shows the realization of a measuring and information system whose application is in facilities such as greenhouses and aims to measure the physical values, achieve wireless transmission, processing and display of data, as well as the ability to automate the operation of devices located in the building, such as water pumps, valves on hoses, heaters, and similar devices that can be found in the building. The system is designed as a "low cost" solution and uses widely available devices and sensors.*

Keywords *SHT35, SHT31, TEMT6000, NRF24L01, Arduino Nano, Raspberry Pi, Python*

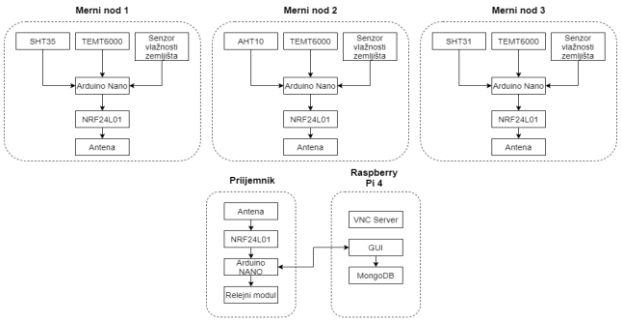
1. UVOD

U današnjem vremenu možemo da primetimo da tehnologija pronalazi primenu u sistemima koji se primenjuju u poljoprivredi a za cilj imaju automatizuju procesa koji se odvijaju.

Sistem koji je projektovan i koji će biti prikazan u ovom radu ima za cilj da pomogne u automatizaciji procesa uzgajanja voća i povrća u stakleniku ili plasteniku.

Automatizacija je sastoji iz merenja vrednosti veličina koje su unapred definisane i regulacije rada uređaja na osnovu izmerenih vrednosti. Veličine koje se mere su temperatura, relativna vlažnosti vazduha, vlažnost zemljišta i nivo osvetljenja.

Uređaji kojima se upravlja su pumpa za vodu, tri ventila, grejalica, osvetljenje i ovlaživač vazduha. Na slici 1 data je blok šema celog sistema.



Slika 1: Blok šema projekta

2. ANALIZA DELOVA SISTEMA

2.1. Merni nod

Merni nod se sastoji od Arduino Nano sistema [1] i NRF24L01 [2] bežičnog modula za prenos podataka. Na slici 2 nalazi se prikaz NRF24L01 bežičnog modula.

Slika 2: NRF24L01 bežični modul

R.B.	Ime senzora	Opseg temperature (°C)	Opseg r.v. vazduha (%)	Komunikacija	Napajanje
1	SHT35	-40 – 125 (0.015 +/- 0.2)	0 – 100 (0.01 +/- 1.5)	I2C	2.4 V – 5.5 V
2	SHT31	-40 – 125 (0.015 +/- 0.3)	0 – 100 (0.01 +/- 2)	I2C	2.4 V – 5.5 V
3	SHT30	-40 – 125 (0.015 +/- 0.3)	0 – 100 (0.01 +/- 3)	I2C	2.4 V – 5.5 V
4	AHT10	-40 – 85 (0.01 +/- 0.3)	0 – 100 (0.024 +/- 2)	I2C	3.3 V – 5.0 V
5	DHT22	-40 – 80 (0.1 +/- 0.5)	0 – 100 (0.1 +/- 2)	One – Wire	3.3 V – 6.0 V
6	AM2320	-40 – 80 (0.1 +/- 0.5)	0 – 99.9 (0.1 +/- 3)	I2C	3.1 V – 5.5 V

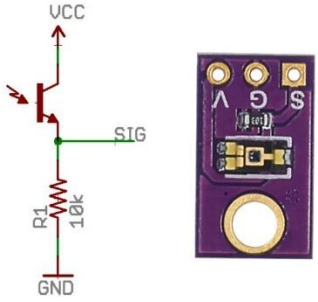
Tabela 1: Specifikacije senzora za merenje temperature i relativne vlažnosti vazduha [3][4][5][6]

Napajanje se vrši pomoću baterije od 9 V a sa dva naponska regulatora postižemo potrebne naponske nivoe od 5 V potrebnog za napajanje mikrokontrolera i senzora i

3.3 V potrebnog za napajanje bežičnog modula. Na sam Arduino su povezani senzori SHT35 [3], SHT31 [3] i AHT10 [4] za merenje temperature i vlažnosti vazduha koji sa Arduinom komunicira preko I2C (*Inter-Integrated Circuit*) protokola. U tabeli 1 su date osnovne karakteristike senzora za temperaturu i relativnu vlažnost vazduha. Od zavisnosti od zadatih specifikacija od strane korisnika može se birati senzor ili senzori koji se koriste u sistemu. Kao senzor za merenje nivoa osvetljenosti koristi se senzor TEMT6000 [7] analognog izlaza. Opseg

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Platon Sovilj.

merenja nivoa osvetljenosti je od 10 lux do 1000 lux a napaja se sa 3.3 V. Sam senzor je realizovan kao naponski razdelnik, a njegov izgled se može videti na slici 3.



Slika 3: TEMT6000 šema (levo) i izgled (desno) senzora

Kao senzor vlažnosti zemljišta koristi se kapacitivni senzor [8] koji daje informaciju preko analognog izlaza u opsegu od 0 V do 3.3 V.

Napajanje se vrši sa 3.3V a na slici 4 prikazan je izgled senzora u verziji 1.2.

Slika 4: Kapacitivni senzor vlažnosti zemljišta (v1.2)

Na svakih 10 sekundi vrši se akvizicija podataka sa senzora, obrada i slanje paketa podataka koji se formatira na sledeći način:

nodeID_temp_hum_soil_lux_batlvl

gde je:

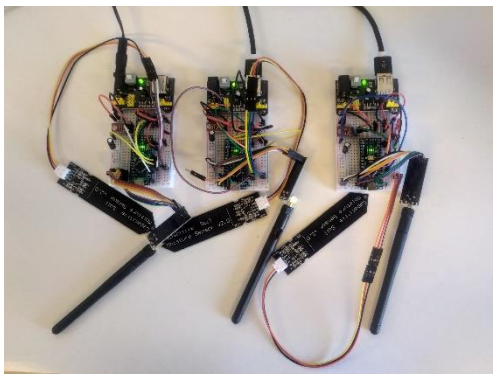
- *nodeID* – ID noda koji šalje podatke
- *temp* – očitana temperatura
- *hum* – očitana vlažnost vazduha
- *soil* – očitana vlažnost zemljišta
- *lux* – očitan nivo osvetljenja
- *batlvl* – preostali procenat baterije

Posle kreiranja paketa podataka koji je potrebno poslati, kreirani paket se prosleđuje bežičnom modulu putem SPI (*Serial Peripheral Interface*) komunikacionog protokola. Komunikacija između prijemnika i predajnika se odvija na 2.4 GHz, koristeći GMSK (*Gaussian Minimum Shift Keying*) modulaciju.

Nakon slanja paketa mikrokontroler i bežični modul ulaze u *sleep* mod kako bi se smanjilo opterećenje na bateriju i produžilo vreme trajanje jedne baterije. Na slici 5 dat je izgled tri merna noda koji se koriste u sistemu.

2.2. Prijemnik

Prijemnik podataka je realizovan pomoću Arduino Nano mikrokontrolera i NRF24L01 bežičnog modula. Sam bežični modul je ovde podešen da radi kao prijemnik i prikuplja podatke koji mu se šalju sa svih mernih nodova.



Slika 5: Prikaz tri merna noda

Pristigle podatke, preko UART (*Universal asynchronous receiver-transmitter*) konekcije, prosleđuje GUI (*Graphical user interface*) programu koji se nalazi na Raspberry Pi [9] platformi. Pored bežičnog modula, na Arduino Nano je povezan i skup od osam relejnih modula koji služe za upravljanje uređajima koji se nalaze u objektu. Upravljanje ovim uređajima se vrši iz GUI-a, koji šalje, preko UART-a, stanja svakog uređaja. Na osnovu poslate poruke mikrokontroler postavlja relejne module u zadata stanja preo svojih digitalnih izlaza. Dodatno, u kod mikrokontrolera je dodata funkcionalnost koja proverava da li je GUI povezan na TX i RX linije tako što na svakih 5 sekundi pošalje poruku kojom proverava prisutnost na drugoj strani. Ukoliko ne dobije odgovor u roku od 10 sekundi, mikrokontroler će postaviti sve uređaje u neaktivno stanje, kako bi sprečio dalji rad bez nadzora. Ova funkcionalnost je korisna u slučaju da se, neočekivano, zatvori program kojim se upravlja ovim sistemom.

2.3. Raspberry Pi 4

Kao centralni deo sistema koristi se Raspberry Pi 4 računar na kome se nalazi GUI aplikacija kojom se upravlja čitavim sistemom i koja će biti opisana u nastavku rada. Sam računar poseduje Broadcom BCM2711 Quad core Cortex-A72 64bit 1.5 GHz procesor [10] i 4 GB RAM memorije. Operativni sistem kao i svi potrebni fajlovi se nalaze na SD kartici od 16 GB. Napajanje se vrši pomoću punjača koji je napravljen za potrebe ovog računara a koji daje 5 VDC/ 3A na svom izlazu. Pored aplikacije, na RPi se nalazi i MongoDB [11] baza podataka u kojoj se čuvaju svi podaci koji su pristigli od početka rada aplikacije i VNC (*Virtual Network Computing*) [12] koji omogućava korisniku da pristupi RPi računaru sa personalnog računara ili mobilnog telefona. Ovakav način pristupa daje korisniku veću slobodu pri korišćenju celokupnog sistema. Bitno je napomenuti da ovakav pristup zahteva da oba računara budu povezana na istu LAN (*Local Area Network*) i poznavanje IP adrese RPi računara i korisničke lozinke za pristup.

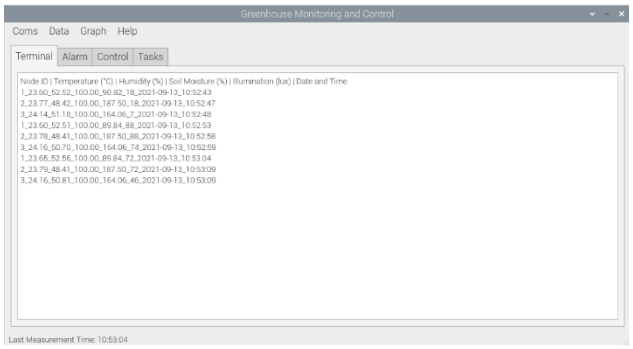
3. GUI APLIKACIJA

Kao što je već rečeno u prethodnom odeljku, na RPi računaru postoji GUI aplikacija kojom se upravlja čitavim sistemom. Sama aplikacija je napisana u Python programskom jeziku i dizajnirana pomoću Qt Designer [13] aplikacije.

Izgled aplikacije podeljen je u četiri celine, odnosno taba. Zadatak aplikacije jeste da prihvata pristigle podatke, obrađuje ih, prikazuje i smešta u bazu podataka. U nastavku rada će biti opisane funkcionalnosti koje se nalaze u svakom tabu.

3.1. Terminal tab

U *Terminal* tabu možemo da pravimo sve poruke koje aplikacija može da prikaže korisniku. To su poruke o novom pristiglom paketu očitanih vrednosti sa nekog mernog noda, uspešnom ili neuspešnom uspostavljanju i započinjanju komunikacije sa prijemnikom, rezultatu postavljanja stanja uređaja i greškama koje mogu da se dogode tokom rada aplikacije. Na slici 6 prikazan je izgled *Terminal* taba.



Slika 6: Izgled Terminal taba

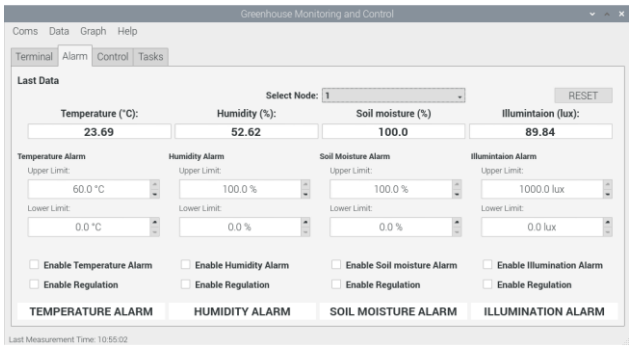
3.2. Alarm tab

U *Alarm* tabu možemo da pratimo poslednje očitane vrednosti sa svakog noda i da, na osnovu tih vrednosti, postavimo alarme za svaku veličinu.

Postavljanje alarma vrši se tako što se izaberu gornja i donja granica za alarm. Ukoliko se poslednja očitana vrednost nalazi van granica zadatog opsega alarm će se aktivirati i polje će biti crvene boje. Ukoliko je poslednja očitana vrednost unutar zadatih granica polje će biti zelene boje.

Ukoliko alarm nije postavljen polje će biti bele boje. Pored mogućnosti postavljanja alarma postoji i mogućnost postavljanja regulacije uređaja na osnovi poslednje očitane vrednosti. Sama regulacije je histerezisnog tipa a za granice histerezisa uzimaju se vrednosti iz polja za granice alarma.

Bitno je napomenuti da u jednom trenutku regulacija može da se obavlja samo na osnovu očitavanja sa jednog mernog noda i da nije moguće postaviti regulaciju na osnovu podataka sa više nodova. Na slici 7 dat je prikaz *Alarm* taba.



Slika 7: Izgled Alarm taba

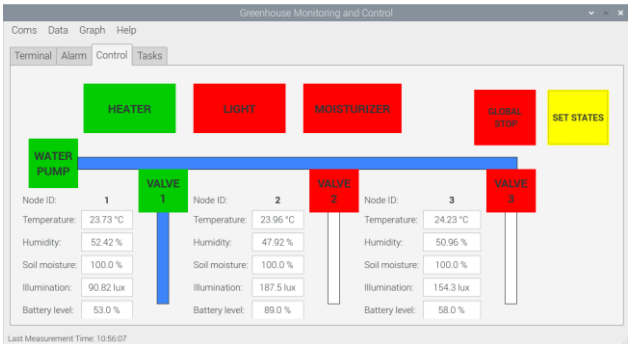
3.3. Control tab

U *Control* tabu imamo mogućnost ručnog postavljanja stanja svakog od uređaja. Stanje uređaja može biti uključen (zelena boja) ili isključen (crvena boja). Kada korisnik odabere stanja svih uređaja potrebno je da izabere opciju *SET STATES* koja prosleđuje sva stanja mikrokontroleru koji daje postavlja releje u zadate pozicije.

Dodatno, kako bi se omogućilo lakše praćenje poslednjih pristiglih podataka, u odeljcima za svaki merni nod prikazuju se poslednji očitani podaci. Ukoliko je potrebno trenutno zaustavljanje svih uređaja to se može učiniti pritiskom na taster *GLOBAL STOP*.

Dodatno, postavljena je svojevrsna zaštita od pogrešnog rukovanja a koja se aktivira kada korisnik greškom želi da uključi pumpu za vodu a ne otvori ni jedan ventil. U tom trenutku se ispisuje poruka upozorenja i pumpa se neće uključiti.

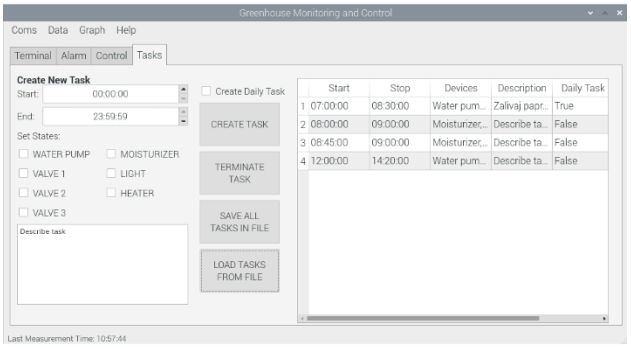
Ukoliko je u tom pokušaju bilo još uređaja kojima se menja stanje to će se uspešno izvršiti. Na slici 8 prikazan je izgled *Control* taba.



Slika 8: Izgled Control taba

3.4. Tasks tab

U *Tasks* tabu korisniku je omogućeno postavljanje zadataka koji treba da se izvrše u toku rada aplikacije. Korisnik može da bira vreme početka i kraja zadatka, stanja svih uređaja tokom trajanja zadatka, pridoda opis zadatka i omogući da se zadatak izvršava svakog dana, u suprotnom on će se izvršiti samo jednom. U tabeli korisnik može jasno da prati sve zadatke koji su postavljeni i da li je i koji zadatak aktivan. Po završetku zadatak, ukoliko se zadatak ne ponavlja i sledeći dan, biće uklonjen iz tabele. Dodatno, korisniku je omogućeno da zadatke koje je postavio sačuva u jedan fajl i da ih iskoristi po potrebi. Na slici 9 dat je izgled *Tasks* taba.



Slika 9: Izgled Tasks taba

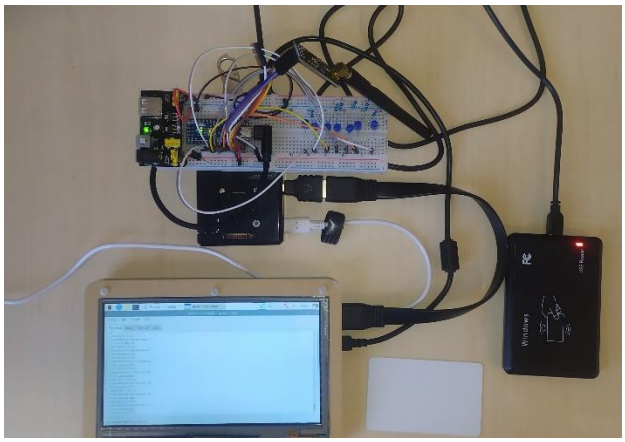
3.5. Dodatne opcije

Kako bi se onemogućio pristup aplikaciji od strane neovlašćenih lica implementiran je sigurnosni sistem koji se zasniva na RFID (*Radio-frequency identicifacion*) karticama i čitaču za iste. Na slici 10 dat je izgled čitača koji se koristi.

Slika 10: RDIF čitač kartica

Kartice koje se koriste rade na frekvenciji od 125 kHz i na sebi nose već predefinisan kod. U samoj aplikaciji su već implementirani kodovi koji su povezani sa nivoom pristupa koji taj kod omogućava. Postoje dva nivoa pristupa, *user* i *master*. Razlika između dva profila korisnika je u tome što *master* korisnik ima mogućnost logovanja podataka iz baze podataka. Prilikom pokretanja aplikacija je zaključana pa je potrebno očitati karticu korisnika koji pristupa sistemu. Ukoliko, posle tri pokušaja, korisnik ne uspe da očita svoju karticu aplikacija ostaje zaključana a samo *master* korisnik može da otključa aplikaciju koristeći svoju karticu.

U padajućim menijima nalaze se dodatne opcije koje nudi ova aplikacija. To su opcije za podešavanje i pokretanje komunikacije sa prijemnikom, grafički prikaz istorije merenja sa mernog noda, kreiranje log datoteke u koju će biti upisani svi podaci koje se nalaze u izabranom vremenskom opsegu logovanja, brisanje kompletne baze podataka i otvaranje About prozora u kom se nalazi više informacija o samoj aplikaciji. Na slici 11 prikazan je izgled kompletnog sistema.



Slika 11: Izgled kompletnog sistema

3. ZAKLJUČAK

U radu je dato rešenje sistema za merenje i regulaciju i stakleniku ili plasteniku.

Sam sistem je uspešno realizovan i zadovoljava potrebe koje su na početku definisane. Sam sistem može da se proširi kako bi imao više od tri merna noda ali treba uzeti u obzir brzinu obrade jednog paketa podataka kako ne bi dolazilo do zakrčenja podataka.

U slučaju veće mreže mernih nodova potrebno je razmotriti i opciju konfiguracije mreže u *mesh grid* infrastrukturu kako bi se optimizovao put podataka od mernog noda do prijemnika. U daljem razvoju sistema potrebno je projektovati PCB (*Printed Circuit Board*) za svaki merni nod i prijemnik. Sledeći korak jeste razvoj sistema do komercijalnog sistema koji je spreman za upotrebu.

4. LITERATURA

[1] https://www.arduino.cc/en/pmwiki.php?n=Main/ArduinoBoardNano

[2] https://cdn.sparkfun.com/datasheets/Wireless/Nordic/nRF24L01_Product_Specification_v2_0.pdf

[3] https://pdf1.alldatasheet.com/datasheet-pdf/view/897976/ETC2/SHT35.html

[4] https://www.handsontec.com/dataspecs/sensor/AHT10.pdf

[5] https://pdf1.alldatasheet.com/datasheet-pdf/view/1132459/ETC2/DHT22.html

[6] https://datasheetspdf.com/pdf-file/952504/Aosong/AM2320/1

[7] https://www.sparkfun.com/datasheets/Sensors/Imaging/TEMT6000.pdf

[8] https://media.digikey.com/pdf/Data%20Sheets/DFRobot%20PDFs/SEN0193_Web.pdf

[9] https://www.raspberrypi.org/products/raspberry-pi-4-model-b/

[10] https://pdf1.alldatasheet.com/datasheet-pdf/view/1283902/ETC1/BCM2711.html

[11] https://www.mongodb.com

[12] https://www.realvnc.com/en/connect/

[13] https://doc.qt.io/qt-5/qtdesigner-manual.html

Kratka biografija:

Milan Šaš rođen je u Beogradu 1997. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Merenje i regulacija odbranio je 2020.god.
kontakt: milansas@uns.ac.rs

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.42:004.738.12
DOI: https://doi.org/10.24867/15BE32Mirilovic

SISTEM ZA SKUPLJANJE I POSTAVLJANJE PRSTENJA SA MAŠINSKIM PREPOZNAVANJEM

RING PICK-AND-PLACE WITH MACHINE VISION

Stefan Mirilović, Milan Vidaković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – *Zadatak rada predstavlja sistem za skupljanje i postavljanje (Pick-and-Place) prstenja sa mašinskim prepoznavanjem. Softverski deo predstavljaju jedna WPF aplikacija, Halcon i Codesys. Hardverski deo predstavlja Fanuc robot CRX-10iA/L sa specijalno napravljenom hvataljkom i kamerom. WPF aplikacija omogućava korisniku da pušta, pauzira i stopira robota, među ostalim funkcijama.*

Ključne reči: *WPF, C#, Codesys, Fanuc Robot, Halcon, Mašinsko prepoznavanje*

Abstract – *This paper deals with a system for pick-and-placing rings using machine vision. Software portion of the system is a WPF application, Halcon and Codesys. Hardware portion is a Fanuc robot CRX-10iA/L with a custom-built gripper and a camera. WPF application allows the user to start, pause and stop the robot, among other functions.*

Keywords: *WPF, C#, Codesys, Fanuc Robot, Halcon, Machine vision*

1. UVOD

Zadatak rada predstavlja razvoj sistema za *pick-and-place* [1] prstenja koji koristi mašinsko prepoznavanje (eng. *Machine vision*) [2]. Ovakvi slični sistemi koriste se u raznim industrijama, od keksova do nakita, da zamene ljudsku radnu snagu i time, dugoročno gledano, smanjuju troškove i čine proizvodnju mnogo produktivnijom.

Za razvoj sistema se morala koristiti kombinacija raznih tehnologija i značajan deo rada je na njihovoj međusobnoj komunikaciji.

2. OPIS HARDVERSKIH TEHNOLOGIJA

Hardverski deo čine robot korporacije *FANUC* [3], model CRX-10iA/L, kamera *Basler* acA1920-25gc i specijalno napravljena hvataljka sa LED osvetljenjem.

2.1. FANUC Robot CRX-10iA/L

FANUC je japanska korporacija koja nudi proizvode i servise u polju automatizacije i najveći je proizvođač industrijskih robota na svetu.

Model CRX-10iA/L je jedan od tzv. kolaborativnih robota (eng. Collaborative robot ili Cobot) [4]. Kolaborativni roboti su predviđeni za bliski rad sa ljudima i zbog toga su u odnosu na uobičajene industrijske robote napravljeni od lakših materijala, imaju glađe ivice, dodatne bezbednosne funkcije, i ograničenja na brzini i sili da bi osigurali bezbednost u radu.

Robot ima širok izbor različitih tipova ulaza i izlaza, od kojih su za ovaj rad najznačajniji digitalni ulazi i izlazi. Robot takođe podržava *Ethernet/IP* [5] konekcije, što će biti ključno za komunikaciju sa softverom.

2.2. Basler acA1920-25gc kamera

Kamera koja se koristi za mašinsko prepoznavanje je Basler acA1920-25gc, rezolucije 1920px x 1080px, 2MP, i može da snima 25 fps (frejmova u sekondi).

Kamera koristi *GigE Vision* [6] interfejs, što je standard uveden 2006. godine za industrijske kamere visokih performansi. On nudi okvir za prenos video zapisa velike brzine preko Ethernet mreže.

2.3. Hvataljka

Na robotu je nameštena specijalno napravljena hvataljka za prstenje. Hvataljka je povezana za dva ventila, ventili za regulator pritiska, a regulator za kompresor za vazduh.

Regulator ograničava maksimalni vezdušni pritisak na zadatu vrednost, koja je nameštena na 0.1 MPa ili 1 bar. Dva ventila kontrolišu otvaranje i zatvaranje hvataljke, zaviseći od toga koji je od njih otvoren.

2.4. Podloga

Za najbolje rezultate mašinskog prepoznavanja mora se odabrati pogodna podloga za objekat koji se prepoznaje. Za prstenje je iznenađujuće najbolji izbor najobičniji blok papir za crtanje. Blok papir pati od bljeska i ne daje homogenu podlogu na slici, ali veliki kontrast bele pozadine i tamnih prstenja je prevladao i davao ubedljivo najbolje rezultate.

3. OPIS SOFTVERSKIH TEHNOLOGIJA

Za softverski deo rada su korišćeni *CodeSys* [7] i *WPF* (*Windows Presentation Foundation*) [8] aplikacija. Za mašinsko prepoznavanje je korišćen *Halcon [9]*.

Za komunikaciju sa robotom je korišćen Ethernet/IP protokol.

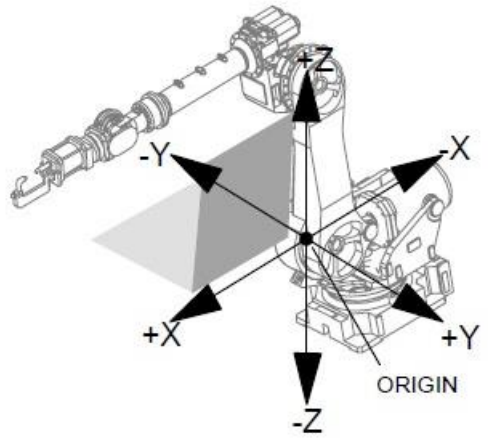
NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Milan Vidaković, red. prof.

3.1. Robotski softver

Robot ima dva suštinski različita koordinatna sistema po kojim može da se pomera.
Jedan koordinatni sistem je Kartezijev (Dekartov) koordinatni sistem od šest osa:

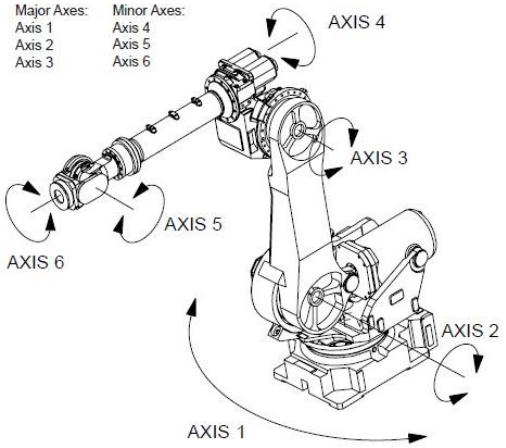
- x – dužina,
- y – širina,
- z – visina,
- w – ugao rotacije oko x-ose,
- p – ugao rotacije oko y-ose i
- r – ugao rotacije oko z-ose.

Slikovno objašnjenje kartezijevog koordinatnog sistema na robotu može se videti na Slici 1.



Slika 1. *Kartezijev koordinatni sistem*

Drugi koordinatni sistem se zove zglobni (eng. Joint). Kao što naziv sugeriše, ose ovog koordinatnog sistema su zglobovi robota i pošto model CRX-10iA/L ima šest zglobova, tako i zglobni koordinatni sistem ima šest osa: J1, J2, J3, J4, J5 i J6. Slikovno objašnjenje može se videti na Slici 2.



Slika 2. *Zglobni koordinatni sistem*

3.2. Codesys

Codesys je razvojno okruženje za programiranje kontrolerskih aplikacija po međunarodnom industrijskom standardu IEC 61131-3 [10]. Ove aplikacije rade na tzv. *PLC*-ovima (eng. *Programmable logic controller*) [11]. PLC je industrijski kompjuter koji je robustan i prilagođen za kontrolu proizvodnih procesa, kao što su montažne trake, mašine i roboti.

Codesys ima mogućnost da pretvori redovan računar u PLC sa njegovim *Codesys Control Win V3*. Ovaj alat je takođe Codesys-ov sistem izvršavanja ili rantajm sistem (eng. *Runtime system*) [12]. Rantajm sistem je neophodan za pokretanje koda napisanog u Codesys-u, jer Codesys ne generiše izvršne programe, kao što je .exe fajl, već zavisi od rantajm sistema da pokrene rantajm okruženje (eng. *Runtime environment*) u kojem se onda izvršava kod.

Po IEC 61131-3 standardu, Codesys nudi pet programskih jezika u svom razvojnom okruženju:

- IL (eng. *instruction list*) je programski jezik sličan asemblerskom, zastareo,
- ST (eng. *structured text*) je slično programiranju u *C*-u ili *Pascal*-u,
- LD (eng. *ladder diagram*) je dijagram koji omogućava programeru da virtualno spoji relejne kontakte i zavojnice,
- FBD (eng. *function block diagram*) je dijagram koji omogućava korisniku da brzinski programira boolean i analogne izraze i
- SFC (eng. *sequential function chart*) je dijagram pogodan za programiranje sekvencijalnih procesa i tokova.

3.3. Windows Presentation Foundation

Windows Presentation Foundation (WPF) je besplatan i open-source grafički podsistem za pravljenje desktop aplikacija na Windows-u. U WPF-u se koristi *Extensible Application Markup Language* (*XAML*) za definisanje korisničkog interfejsa (frontend), i C# za programiranje funkcionalnosti (backend).

3.4. Halcon

MVTec Halcon je obiman softver za mašinsko prepoznavanje sa integrisanim razvojnim okruženjem (*HDevelop*). U smislu funkcionalnosti se može porediti sa *OpenCV* [13] bibliotekom. Verovatno najveća razlika između njih je to što je OpenCV besplatan za korišćenje i open-source, dok Halcon nije ni jedno ni drugo.

Halcon može da se koristi iz njegovog razvojnog okruženja HDevelop, ali može da se poziva njegov API iz *C*-a, *C++*-a, *Visual Basic*-a i C#-a.

4. OPIS IMPLEMENTACIJE

Sistem čine WPF aplikacija i Codesys kontrolerska aplikacija, koji međusobno komuniciraju preko deljene memorije.

4.1. Komunikacija sa Codesys-om

Za komunikaciju sa Codesys-om, WPF aplikacija ima dve promenljive: inputVariables i outputVariables. InputVariables je struktura koja sadrži sve ulazne

parametre, tj. parametre koje idu iz pravca Codesys u WPF. OutputVaribles je struktura koja sadrži sve izlazne parametre, iz pravca WPF u Codesys.

Komunikacija između WPF aplikacije i Codesys-a je uspostavljena preko deljene memorije, koja se kontinualno čita i piše u beskonačnoj petlji, koja se poziva preko BackgroundWorker klase. BackgroundWorker klasa izvršava operacije na zasebnoj niti, što dozvoljava spomenutoj petlji da se vrti kontinualno i asinhrono bez ometanja glavne niti.

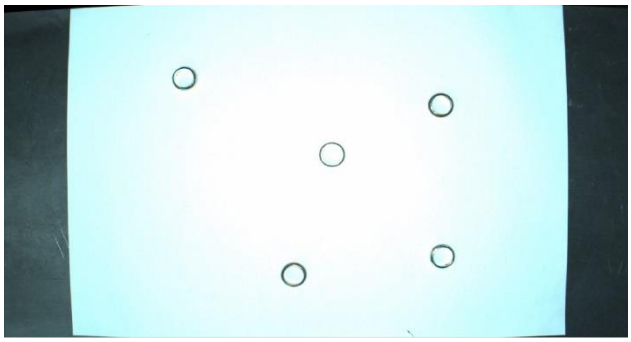
Radi komunikacije se prave dva *Memory-mapped* fajla. Oni su neperzistentni fajlovi koji nisu povezani sa datotekom na disku.

Kada poslednji proces završi rad sa datotekom, podaci se gube. Ovaj vid komunikacije je odabran zbog njegove odlične brzine.

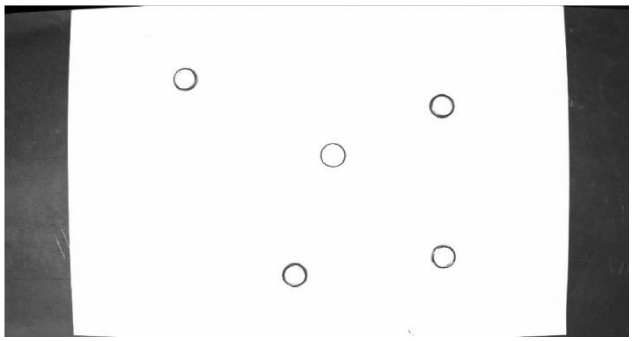
4.2. Prepoznavanje prstenja

Kao što je već rečeno, za prepoznavanje se koristi Halcon-ov API. Sa operacijom Decompose3 trokanalna (RGB, eng. *Red, Green, Blue*) slika koju kamera uslika se pretvori u tri jednokanalne slike, gde svaka odgovara jednoj boji od tri koje zajedno čine RGB. Originalna slika i jedna generisana se vide na Slikama 3. i 4.

Najveći kontrast između pozadine i prstenja davala je slika koja odgovara plavoj boji, pa se ona koristila dalje u programu.



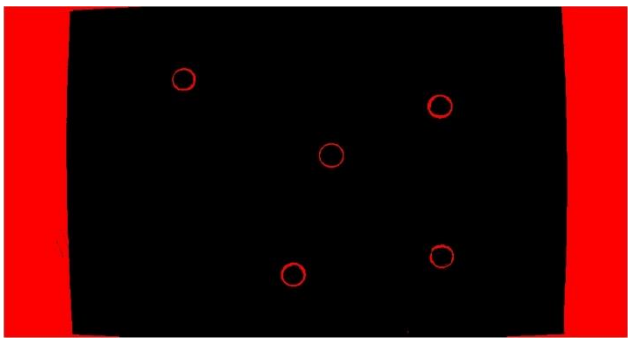
Slika 3. *Originalna slika*



Slika 4. *Izdvojena plava boja iz slike*

Sledeća operacija je *Threshold*. Njen zadatak je izdvojiti sve piksele čija je vrednost sive boje između 0 i 150 **(**0 je crna boja, a 255 bela**)**.

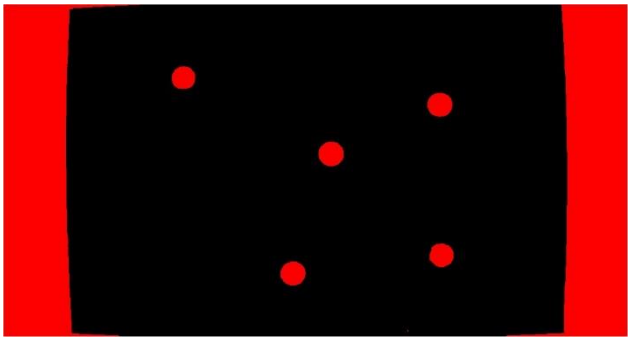
Praktično, ovaj korak je uklonio belu podlogu, a ostatak označio, kao što se vidi na Slici 5. Programski, sve boje slike su se pretvorile u nule i jedinice, gde su nule predstavljene crnom, a jedinice crvenom na Slici 5.



Slika 5. *Izdvojene tamne površine*

Sledeća operacija *Connection* čini da svaki region koji nije spojen sa drugim bude odvojena komponenta.

Ovo će kasnije omogućiti brojanje komponenti, tj. prstenja. Sledeća operacija *FillUp* popuni sve komponente koji su šuplje. Time se dobije Slika 6.



Slika 6. *Popunjene šupljine*

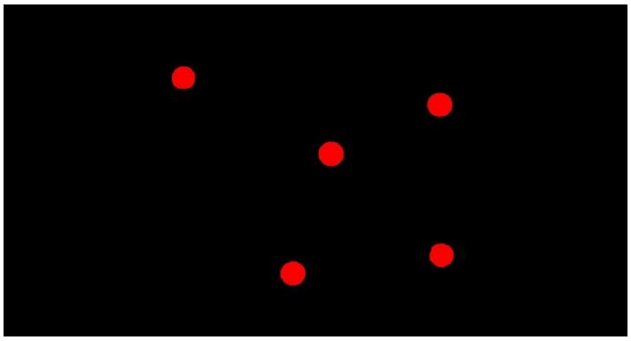
Sledeća operacija *SelectShape* čini više stvari; za svaku komponentu izračuna dve vrednosti:

- *circularity* – procenat koji predstavlja koliko se površina komponente poklapa sa krugom i
- *area* – vrednost površine komponente.

Kad ima ove dve vrednosti za sve komponente, izdvoji sve komponente koji zadovoljavaju sledeće uslove:

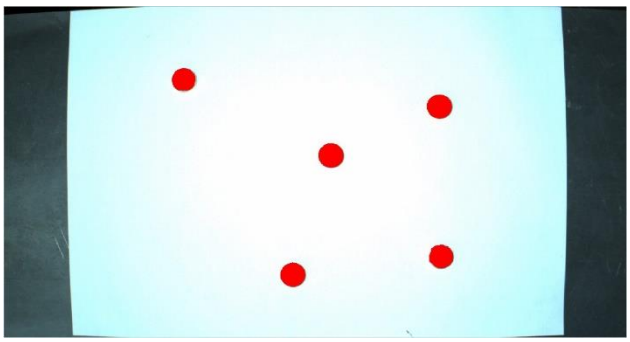
- 0.85 < *circularity* < 1 i
- 400 < *area* < 9999.

Rezultat ove operacije vidi se na Slici 7.



Slika 7. *Rezultat SelectShape operacije*

Potom iscrta na prozoru dobijeni rezultat sa Slike 7, što onda izgleda kao na Slici 8.



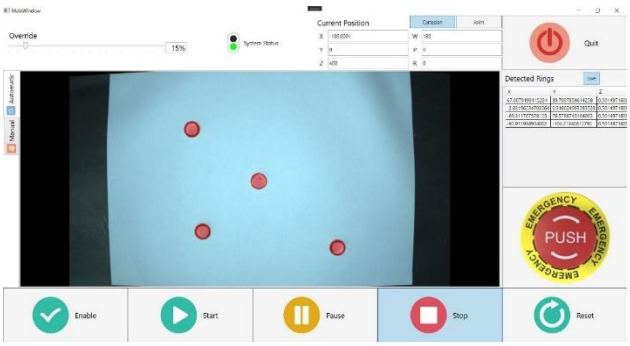
Slika 8. *Konačni rezultat*

Ovim je trebalo biti prepoznato svo prstenje, gde svaki prsten odgovara jednoj komponenti. Sledeći korak je onda prebrojati komponente sa funkcijom CountObj. Potom se traži *row* i *column* svake komponente, što predstavlja njegovu poziciju na slici. U gornjem levom uglu slike je row 0 i column 0, a u donjem desnom je row 1080, a column 1920, što odgovara rezoluciji kamere (1920x1080 piksela). U ovakvom formatu, row i column su robotu neupotrebljivi, zato ih treba pretvoriti u x i y koordinate po robotskom koordinatnom sistemu.

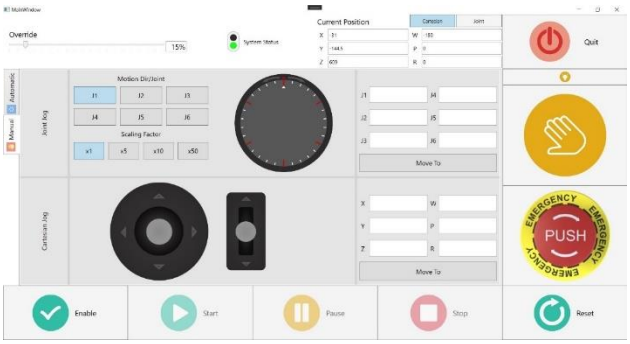
Nad svim koordinatama se onda vrši operacija *ImagePointsToWorldPlane* koja vrši potrebnu konverziju i daje x i y koordinate u milimetrima. Od ovih koordinata se napravi pozicija koja se onda dodaje na listu pozicija koja se šalje Codesys-u.

4.3. Korisnički interfejs

Korisnički interfejs se sastoji od samo jednog prozora u dva režima. Režim menja mnogo elemenata interfejsa, automatski režim može se videti na Slici 9, a ručni na Slici 10.



Slika 9. *Korisnički interfejs*



Slika 10. *Korisnički interfejs u ručnom režimu*

5. ZAKLJUČAK

Sistem je razvijen u aktuelnim tehnologijama i sa modelom robota iz najnovije generacije kolaborativnih robota, koji su tek ušli u proizvodnju. Sistem u trenutnom stanju nema neku veliku praktičnu korist, ali daljim razvojem bi se to moglo lako promeniti. Jedan vid daljeg razvoja bi bila zamena statične podloge sa pokretnom trakom. Još jedna ideja za dalji razvoj bi bilo uključivanje i nekakve obrade prstenja nakon skupljanja, poput poliranja.

6. LITERATURA

[1] Pick-and-place
https://en.wikipedia.org/wiki/Pick-and-place_machine
[2] Machine vision
https://en.wikipedia.org/wiki/Machine_vision
[3] FANUC
https://en.wikipedia.org/wiki/FANUC
[4] Collaborative robot
https://en.wikipedia.org/wiki/Cobot
[5] Ethernet/IP
https://en.wikipedia.org/wiki/EtherNet/IP
[6] GigE Vision
https://en.wikipedia.org/wiki/GigE_Vision
[7] Codesys
https://en.wikipedia.org/wiki/CODESYS
[8] Windows Presentation Foundation
https://en.wikipedia.org/wiki/Windows_Presentation_Foundation
[9] Halcon https://www.mvtec.com/products/halcon
[10] IEC 61131-3
https://en.wikipedia.org/wiki/IEC_61131-3
[11] Programmable logic controller
https://en.wikipedia.org/wiki/Programmable_logic_controller
[12] Runtime system
https://en.wikipedia.org/wiki/Runtime_system
[13] OpenCV
https://en.wikipedia.org/wiki/OpenCV

Kratka biografija:

Stefan Milivović rođen je 04.05.1997. u Bečeju. Završio osnovnu školu „Zdravko Gložanski“ u Bečeju 2012. godine i Gimnaziju Bečej 2016. godine. godine. Završio osnovne akademske studije na Fakultetu tehničkih nauka u Novom Sadu, smer Računarstvo i automatika 2020. godine. Nakon završenih osnovnih studija upisao je master akademske studije na istom fakultetu, smer Računarstvo i automatika, modul Elektronsko poslovanje.
kontakt: stefan.mirilovic@gmail.com

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

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.738.1:766
DOI: https://doi.org/10.24867/15BE33Nikolic

ANALIZA GRAPHQL I REST API-IJA ZA INFORMACIJE O AKADEMSKIM ENTITETIMA

ANALYSIS OF GRAPHQL AND REST API IN INFORMATION RETRIEVAL OF ACADEMIC ENTITIES

Dušan Nikolić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – PRIMENJENE RAČUNARSKE NAUKE I INFORMATIKA

Kratak sadržaj – *U radu su analizirane performanse između GraphQL i REST API-ija prilikom pretrage informacionog sistema naučno-istraživačke delatnosti. Preuzeti su podaci o akademskim entitetima sa Elsevier Scopus platforme. Prikazana je arhitektura informacionog sistema sa dijagramom razmeštaja i dijagramom komponenti. Implementiran je GraphQL API i diskutovani su rezultati odziva prilikom pretrage datih entiteta u oba API-ija.*

Ključne reči: *informacioni sistemi naučno-istraživačke delatnosti, migracija, GraphQL, REST.*

Abstract – *This paper analyzes performance differences between GraphQL and REST API's information retreival of a current research information system. Data regarding academic entities has been taken from the Elsevier Scopus platform. Information system architecture is shown with deployment and component diagrams. In addition to recording the implementation of the GraphQL API, the present study also discusses the results of system response gathered during information retrieval of the given entities in both APIs.*

Keywords: *current research information systems, migration study, GraphQL, REST.*

1. UVOD

Predmet istraživanja ovog rada jeste analiza performansi između GraphQL i REST API-ija prilikom pretrage akademskih entiteta informacionog sistema naučno-istraživačke delatnosti. U radu je opisan proces migracije API-ija sa tradicionalnog REST stila na GraphQL.

Kako bi vršenje upita nad informacionim sistemom bilo moguće, preuzeto je preko 13 hiljada informacija o akademskim entitetima sa Elsevier *Scopus* platforme. Preuzete informacije su pretežno publikacije u časopisu, monografije radova i radovi objavljeni u zbornicima sa naučno-stručnih konferencija koje pripadaju autorima sa Univerziteta u Novom Sadu.

1.1. Informacioni sistemi naučno-istraživačke delatnosti

Kroz upotrebu informacionih sistema naučno-istraživačke delatnosti (eng. *Current Research Information Systems – CRIS*), institucije koje se bave naukom dobijaju sveobuhvatan pogled na aktivnosti i delatnosti datih istraživača u objedinjenoj bazi podataka. Ovakve aktivnosti obuhvataju proces prikupljanja, obrade i upravljanja informacijama o ljudima, publikacijama, patentima, tezama, opremi i projektima od istraživačkog značaja [1].

Postoje informacioni sistemi koji korisnicima olakšavaju proces pretrage citata i publikacija. Scopus, razvijen od strane *Elsevier* kompanije, predstavlja bazu podataka (eng. *abstract and citation database*) sa preko 75 miliona indeksiranih apstrakata i citata koji su dobijeni analizom publikacija, monografija radova i radova sa naučno-stručne konferencije objavljenih nakon 1966. godine [2] Funkcionalnosti koje Scopus nudi su: osnovna pretraga, pretraga po autoru, napredna pretraga, brza pretraga.

1.2. REST

REST (eng. *Representational State Transfer*) predstavlja stil softverske arhitekture namenjen distribuiranim sistemima. Stil arhitekture specificira ograničenja kao što su: način identifikacije resursa, uniforman interfejs sa određenom semantikom, samoopisive poruke, *stateless* interakcija.

API protokoli su se menjali i razvijali tokom vremena. Uzimajući u obzir sve veću dostupnost podataka, potražnja za fleksibilnijim i javno dostupnim API-ijima je takođe veća i REST je u industriji trenutno najzastupljeniji API protokol. Međutim, ograničenje ovakvog protokola je u tome što pozivi ka API-iju uglavnom zahtevaju više povezanih resursa, rezultujući u višestrukim zahtevima ka serveru. Ovakvo ograničenje otežava aplikacijama da se efikasno integrišu sa REST API-ijima, jer integracija podrazumeva pristup ka resursima iz više uzastopnih zahteva.

1.3. GraphQL

GraphQL je upitni jezik otvorenog koda razvijen od strane kompanije *Facebook*. Predstavlja upitni jezik nad API-ijem (eng. *Application Programming Interface*), namenjen za implementiranje veb servisa.

Osnovna karakteristika GraphQL-a je mogućnost kreiranja proizvoljnih upita ka serveru. Upiti su kreirani time što se na serverskoj strani implementira statička šema (eng. *schema*) nad bazom podataka. Moguće je implementirati i dinamičku (eng. *runtime*) šemu. Šema se kreira unapred i predstavlja pogled na bazu podataka,

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Ivanović, red. prof.

odnosno API koji klijenti mogu da pozivaju. Ovakva implementacija upita je kontrast u odnosu na REST pristup. U REST sistemima se programski implementiraju operacije (eng. *endpoints*) koje klijenti mogu da pozovu, dok u GraphQL sistemima klijenti kreiraju upite nad „*export*-ovanom" bazom podataka.

1.4. Pregled relevantne literature

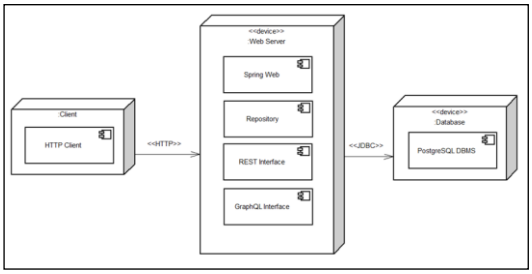
U radu [3] dokumentovan je process migriranja 7 GitHub API-ija sa REST-a na GraphQL. Predmet istraživanja u datom radu je pružanje odgovora na pet pitanja (eng. *research questions)*: 1) glavne karakteristike GraphQL-a i njegove prednosti; 2) nedostaci GraphQL tehnologije; 3) smanjenje broja poziva ka API-iju upotrebom GraphQL-a nasuprot REST pozivima; 4) smanjenje vraćenih polja od strane servera upotrebom GraphQL; 5) smanjenje veličine dokumenta vraćenog od strane servera upotrebom GraphQL. Odgovor na pitanje karakteristika GraphQL-a je dat analizom dokumentacije, ali i blogova koji izveštavaju o novim tehnologijama (eng. *grey literature*). Ističu se dve karakteristike GraphQL-a: hijerarhijski model podataka koji smanjuje potrebu za više *endpoint*-a i mogućnost da klijenti postavljaju proizvoljne upite ka serveru. Za potrebe faze analize performansi GraphQL-a, migrirano je pet GitHub REST projekata otvorenog koda i dva arXiv [4] projekta. Kao rezultat migracije dobijeno je 29 REST API poziva mapiranih na 24 GraphQL upita. Prema dobijenom broju *endpoint*-a, zaključak je da ne postoji značajno smanjenje broja poziva ka serveru upotrebom GraphQL-a Sa druge strane, demonstrirana je značajna razlika u broju polja i veličine dokumenta vraćenih od strane servera ove dve tehnologije. U poređenju sa pozivima REST stila, GraphQL implementacija smanjuje veličinu vraćenog JSON dokumenta za čak 94% u broju vraćenih polja i 99% u broju vraćenih bajtova [3].

Rad [5] prikazuje studiju migriranja dela sistema za upravljanje pametnom kućom na GraphQL API. U izveštaju evaluiraju se performanse na primeru brzine odziva dva *endpoint*-a koja su implementirana u REST i GraphQL. Prvi *endpoint* predstavlja primer poziva koji zahteva mali broj polja od strane servera. Ovakav upit autori nazivaju „atomičan" (eng. *atomic*) i nije uočena razlika u brzini odziva sistema nakon migracije na GraphQL. Drugi *endpoint* je složeniji i zahteva više različitih polja sa servera. Za upit nad drugim *endpoint*-om, GraphQL je 54% brži od REST ekvivalentnog poziva.

Rad [6] iz 2020. godine sprovodi kontrolisan eksperiment sa 22 studenta koji poseduju određen stepen znanja o veb programiranju. Cilj eksperimenta je da studenti implementiraju 8 operacija (*endpoints*) na određenom veb servisu koristeći REST i GraphQL. Rezultati eksperimenta su pokazali da vreme neophodno za implementaciju REST operacije raste sa porastom broja njegovih parametara. Medijana vremena potrebnog za implementaciju REST operacije je 9 minuta, dok za GraphQL iznosi 6 minuta. Kao rezultat, zaključeno je da su GraphQL operacije jednostavnije za studente koji nisu ranije imali susret sa datom tehnologijom. Prema medijanama pokazano je da je studentima potrebno manje truda za implementiranje GraphQL operacije.

2. SPECIFIKACIJA SISTEMA ZA INFORMACIJE O AKADEMSKIM ENTITETIMA

U ovom poglavlju prikazana je arhitektura sistema sa aspekta UML dijagrama razmeštaja i dijagrama komponenti. Dijagram razmeštaja (slika 1.) prikazuje softverske komponente i veze uz pomoć kojih komuniciraju date komponente. Dijagram komponenti na slici 2. prikazuje organizaciju i veze između komponenti aplikacije.

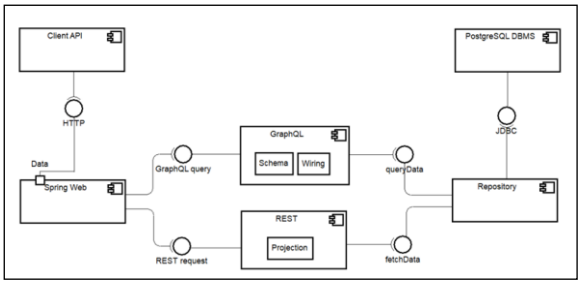


Slika 1. *Dijagram razmeštaja*

Client **–** sadrži komponentu *Http Client*. Komponenta *HTTP Client* može predstavljati proizvoljan veb čitač ili namenski softver za kreiranje HTTP poziva.

Web Server – sadrži komponente *Spring Web*, *Repository*, *REST i GraphQL Interface*.

Database **–** sadrži komponentu *PostgreSQL DBMS* kao izabrani sistem za upravljanje bazom podataka.



Slika 2. *Dijagram komponenti*

Prema dijagramu komponenti sa slike 2, klijentski HTTP zahtev stiže do *Spring Web* komponente, te se dalje usmerava ka GraphQL ili REST komponenti. Za potrebe GraphQL upita, data komponenta je sastavljena od *Schema* i *Wiring* dela (eng. *parts*). Zadatak *Schema* komponente je definisanje polja koja GraphQL API podržava.

Ta polja se kasnije mapiraju na bazu podataka informacionog sistema. Sa druge strane *Wiring* komponenta mapira pristigli klijentski upit na odgovarajući entitet iz baze podataka. REST komponenta se sastoji od *Projections* biblioteke, čiji zadatak je mapiranje korisničkog zahteva na entitet iz *Repository* sloja. Konačno, JPA *Repository* komponenta uz pomoć JDBC interfejsa komunicira sa PostgreSQL bazom podataka.

3. REZULTATI I TUMAČENJA

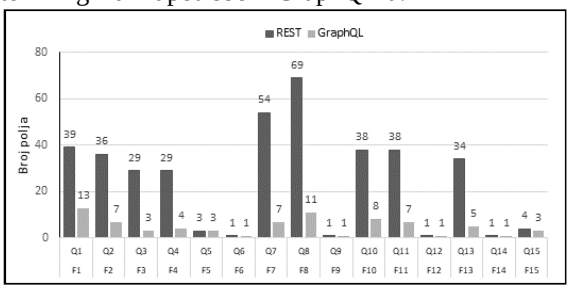
U ovom poglavlju se prikazuju i diskutuju dobijeni rezultati pri analizi performansi prilikom upita nad GraphQL i REST API-ijima.

Kako bi se evaluirale performanse, implementirano je 15 reprezentativnih upita $[Q1, \dots, Q15]$ ka akademskim entitetima u GraphQL tehnologiji i 15 ekvivalentnih REST poziva $[F1, \dots, F15]$.

Odeljak 3.1 poredi broj vraćenih polja u JSON dokumentu oba API-ija i prikazuje medijane u odnosu na 15 GraphQL upita i REST poziva. Odeljak 3.2 poredi veličinu dokumenta vraćenog od strane servera u bajtima.

3.1. Vraćena JSON polja

Kada je broj vraćenih polja u JSON odgovoru u pitanju, rezultati obe tehnologije prikazani su na grafikonu 1. Grafikon prikazuje broj vraćenih polja na pozive iz upita $[Q1, \dots, Q15]$ koji su implementirani uz pomoć REST-a, a potom migrirani upotrebom GraphQL-a.



Grafikon 1. *Broj vraćenih REST i GraphQL polja u JSON odgovoru*

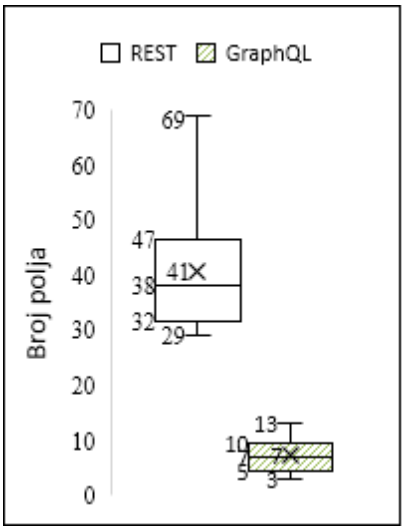
Uočava se znatno manji broj vraćenih polja upotrebom GraphQL-a. Ovaj stepen redukcije se menja u zavisnosti od upita i varira od razlike u 1 polju (F14, Q14) do razlike u 58 polja (F8, Q8). Razlog za znatan REST *overfetch*, na primeru (F8, Q8), je u samoj implementaciji Spring *Projections* tehnologije. Projekcija vraća rezultat entiteta sa svim svojim korenskim poljima. Međutim, uzimajući u obzir povezanost akademskog entiteta koji pripada pozivu (F8) sa većim brojem drugih entiteta, projekcija takođe vraća i povezane entitete. Kao posledica, rezultat vraća JSON sa 38 korenskih i 31 ugnježđenih polja.

Sa druge strane, GraphQL eliminiše *overfetching* time što se unapred znaju polja koja se zahtevaju u telu zahteva. U konkretnom upitu Q8, zahtevaju se 11 nepraznih polja *PaperMonograph* - monografije radova akademskog entiteta koja su prikazana u listingu 1.

```
query Q8 {
findPaperMonographByScopusID(scopusID: "85019996128" ) {
    id
    startPage
    endPage
    titleEng
    abstractEng
    language {
      code
      name
    }
    monographOfPapers {
      id
      editionTitle
    }
  }
}
```

Listing 1. *7 korenskih i 4 ugnježđena polja GraphQL upita*

Grafikon 2 prikazuje *box plot* sa distribucijom broja JSON polja vraćenih od strane REST i GraphQL API-ija. *Box plot* prikazuje uređenu seriju od najmanjeg ka najvećem uzorku i sadrži medijanu i kvartile. Medijana je srednja vrednost po položaju koja deli uzorak na dva jednaka dela. Jedna polovina vrednosti uzorka je manja od medijane, a druga polovina je veća. Kvartili su srednje vrednosti po položaju koje dele uređenu seriju na četiri jednaka dela. Iz grafikona su izuzeti podaci o upitima i i funkcijama koje vraćaju isti broj polja. Medijana broja polja vraćenih sa REST pozivima je 41, dok kod GraphQL poziva je 7. Mera prvog kvartila je 32 (REST) i 5 (GraphQL). Četvrti kvartil iznosi 47 (REST) i 10 (GraphQL).

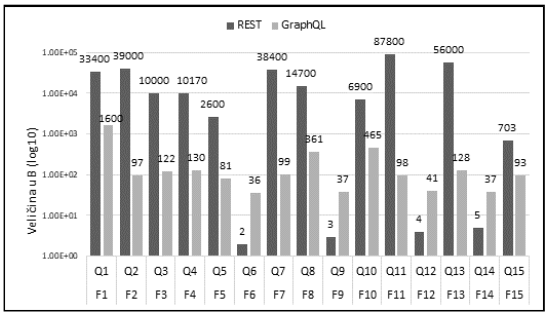


Grafikon 2. *Box plot broja vraćenih REST i GraphQL polja*

3.2. Veličina JSON dokumenata

Nakon što su implementirani GraphQL i REST upiti $[Q1, \dots, Q15]$, izmerena je veličina odgovora JSON odgovora i prikazana na grafikonima 3 i 4. Grafikon 3 prikazuje veličinu JSON dokumenata merenu u bajtima sa log10 poravnanjem. Na vrhu kolona prikazana je labela koja predstavlja izmeren stvaran broj bajtova.

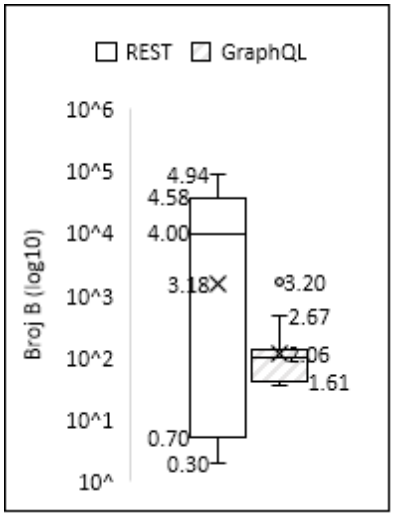
U skoro svim upitima uočava se znatna razlika veličine JSON dokumenta nakon migracije na GraphQL. Na primeru Q8, upotrebom REST poziva odziv je 14.7KB. Isti upit implementiran uz pomoć GraphQL-a iz listinga 11 je 361B. Raskorak je izraženiji ukoliko REST implementacija upita vraća više povezanih entiteta.



Grafikon 3. *Veličina JSON odgovora prilikom REST i GraphQL upita*

Grafikon 4 prikazuje *box plot* sa distribucijom veličine JSON dokumenata vraćenih od strane REST-a i GraphQL. REST odgovori vraćaju 1.51KB (medijana), nasuprot 114B medijani GraphQL odgovora. Samim tim, prema medijani je postignuta memorijska redukcija

GraphQL API-ija oko jednog reda veličine u odnosu na REST. Zbog uticaja atomičnih upita donji REST kvartil je 5B, dok kod GraphQL je 40B. Gornji kvartil je ~38KB (REST) i 128B (GraphQL).



Grafikon 4. *Box plot veličine JSON dokumenata pri REST i GraphQL pozivima*

4. ZAKLJUČAK

Implementiran je GraphQL API kao *wrapper* oko postojećeg REST interfejsa i prilikom postavljanja 15 različitih upita meren je odziv sistema prema dva kriterijuma.

Prvi kriterijum predstavljao je broj vraćenih polja od strane REST i GraphQL API-ija. Kao rezultat, prema medijanama je pokazano da GraphQL vraća do 6 puta manje polja od REST ekvivalentnog poziva.

Drugi kriterijum predstavljao je veličinu dokumenta vraćenog od strane servera. Pokazalo se da GraphQL vraća za red veličine manji JSON dokument (medijana – 114B) od REST API-ija (medijana – 1.51KB). Ovi rezultati pokazuju znatnu uštedu resursa pri svakom zahtevu.

4.1. Dalji razvoj sistema

Uzimajući u obzir da je u ovom radu podržana samo pretraga, jedan od mogućih proširenja u smeru dalje analize performansi GraphQL i REST API-ija bi bila implementacija podrške kreiranja, ažuriranja i brisanja akademskih entiteta na informacionom sistemu. Pomoću GraphQL API-ija takva podrška podrazumeva implementaciju mutacija, dok uz pomoć REST API-ija bi se implementirale dodatne operacije (*endpoints*). Takođe, prema uštedi memorije pri pozivima koji su prikazani u ovom radu, od značajnije koristi bi bila dalja istraživanja merenja performansi GraphQL-a u kontekstu mobilnih i distribuiranih aplikacija. Takvo istraživanje bi utvrdilo da li je moguće dodatno optimizovati resurse pri razmeni podataka na mobilnim i mikroservisnim aplikacijama.

5. LITERATURA

[1] Azeroual, O., & Schöpfel, J. (2019). Quality issues of CRIS data: An exploratory investigation with universities from twelve countries. Publications, 7(1), 14.

[2] https://blog.scopus.com/posts/scopus-roadmap-whats-coming-up-in-2020-2021 (pristupljeno u septembru 2021.).

[3] Brito, G., Mombach, T., & Valente, M. T. (2019). Migrating to GraphQL: A practical assessment. In 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER) (pp. 140-150). IEEE.

[4] https://arxiv.org/ (pregledano u septembru 2021.).

[5] Vogel, M., Weber, S., & Zirpins, C. (2017). Experiences on migrating RESTful web services to GraphQL. In International Conference on Service-Oriented Computing (pp. 283-295). Springer, Cham.

[6] Brito, G., & Valente, M. T. (2020). Rest vs graphql: A controlled experiment. In 2020 IEEE International Conference on Software Architecture (ICSA) (pp. 81-91). IEEE.

Kratka biografija:

Dušan Nikolić rođen je 16.10.1997. godine u Loznici. Smer računarstvo i automatika na Fakultetu tehničkih nauka u Novom Sadu upisao je 2016 godine. Osnovne studije završio je u septembru 2020. godine. Od oktobra 2020. godine upisuje master studije i angažovan je kao saradnik u nastavi.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.317
DOI: https://doi.org/10.24867/15BE34Tesanovic

IOT MERNO-INFORMACIONI SISTEM ZASNOVAN NA FREERTOS OPERATIVNOM SISTEMU

IOT MEASUREMENT AND INFORMATION SYSTEM BASED ON FREERTOS OPERATING SYSTEM

Stefan Tešanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast- ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Ovaj rad predstavlja merno-informacioni sistem koji ima ulogu male meteorološke stanice.*

Ključne reči: *IoT, FreeRTOS, merenje, akvizicija*

Abstract – *This paper presents a measurement and information system that acts as a small meteorological station.*

Keywords: *IoT, FreeRTOS, measurement, acquisition*

1. UVOD

Uticaj IoT (*Internet of things*) se može videti u gotovo svim sferama današnjeg vremena, bilo da je to industrija, medicina, poljoprivreda pa sve do svakodnevne upotrebe u domaćinstvu. Njegova uloga je da se uređaji koji u normalnom slučaju ne komuniciraju spoje i samostalno izrvršavaju radnje, uzimajući u obzir podatke koje dobijaju od drugih uređaja.

Razlozi zbog kojih se sve više koristi ovaj koncept su: sve se dešava u realnom vremenu, mnogo je veća bezbednost korisnika, automatski je i uz manje napora. Ovo su samo neke od prednosti IoT u odnosu na pojedinačne uređaje, ali pored prednosti postoje i mane.

Neke od značajnijih mana su: memorijski proctor (sa povećanjem broja uređaja potrebno je i više mesta za skladištenje svih podataka), sigurnost podataka (potencijalna meta hakera), greške ili bagovi (*bugs*) koji se mogu javiti pri komunikaciji i kompatibilnost uređaja u sistemu.

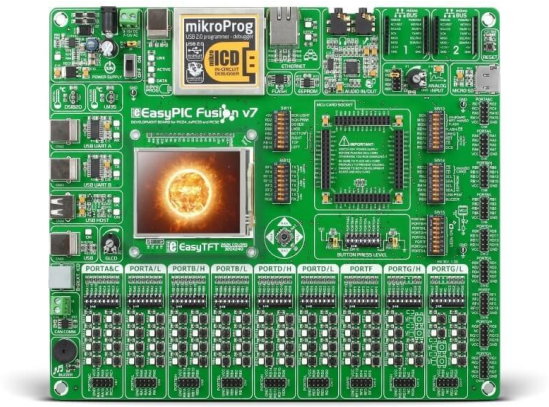
U radu je opisan projekat koji ima ulogu male meteorološke stanice. Sastoji se iz dva dela, hardverskog (četiri senozorska modula, EasyPIC Fusion razvojna ploča sa mikrokontrolerom i računara) i softverskog dela (server, baza podataka i veb stranica).

Senzorski moduli postavljeni su na ploču koja na sebi ima mikrokontroler, a razvojna ploča je povezana sa računarom putem serijske komunikacije. Računar koji ima ulogu servera prima podatke, parsira ih i upisuje u bazu podataka. Podaci se uzimaju iz baze podataka i prikazuju na veb stranici u vidu tabele i grafika.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Platon Sovilj.

2. HARDVER

Ključna komponenta hardverskog dela sistema jeste razvojna ploča EasyPIC Fusion v7, razvijena od strane kompanije Mikroelektronika. Ploča podržava mikroprocesore iz porodica dsPIC33, PIC24 i PIC32, a na sebi poseduje mnoge module.



Sl 1. *Izgled EasyPIC Fusion v7 razvojne ploče*

U projektu su od modula koje ploča poseduje korišćeni mikroBUS™ soketi i UART konektor, mikroprocesor je iz porodice PIC32, odnosno PIC32MZ2048ECH144.

2.1. Senzorski moduli

Projekat ima četiri senzorska modula koji šalju podatke mikrokontroleru. Podaci se mogu primati istovremeno sa više senzora, a korisnik može da odluči na koliko će se sekundi podaci slati mikrokontroleru, koji ih dalje obrađuje. Na slici 2. su prikazani korišćeni senzorski moduli: Illuminance click, CO click, DHT22 click i Barometer click.

Sl 2. *Senzorski moduli*

Illuminance click u sebi poseduje **TSL2561** konvertor svetlosti u digitalnu vrednost sa senzorom dizajniranim da

oponaša ljudsku percerpciju svetlosti. Većinski se koristi kod aplikacija koje mere ambijentnu svetlost. Senzor ima dve fotodiode, jedna koja je osetljiva na ceo spektar svetlosti, a druga na infracrveni spektar. Vidljivi spektar se dobija preko formule. Pločica komunicira sa mikrokontrolerom putem I2C komunikacije, a napaja se naponom od 3.3 V [4].

CO click u sebi nosi **MQ-7** senzor za detekciju CO gasa. Opseg detekcije CO se kreće od 20 ppm do 2000 ppm. Komunicira sa mikrontrolerom preko AN pina i napaja se sa 5 V. Takođe postoji potenciometar za kalibraciju senzora [3].

DHT22 click je pločica za merenje vlažnosti i temperature na kojoj se nalazi istoimeni senzor. Senzor može da detektuje temeperature u opsegu od -40 °C do 80 °C sa preciznošću od pola stepena, dok se relativna vlažnost meri od 0 % do 100 % sa preciznošću od 2 %. Senzor može da se napaja sa 3.3 V ili 5 V [1].

Barometer click je senzor visoke preciznosti sebi ima **LPS25HB** integrisano kolo koje daje 24-bitne vrednosti pritiska i temperature. Opseg za merenje pritiska je od 260 hPa do 1260 hPa. Precinost ovog senzora može biti i do 0.01 hPa ukoliko se nalazi u visoko rezolucionom modu. Napon napajanja senzora je 3.3 V i može da komunicira putem SPI ili I2C komunikacije [2].

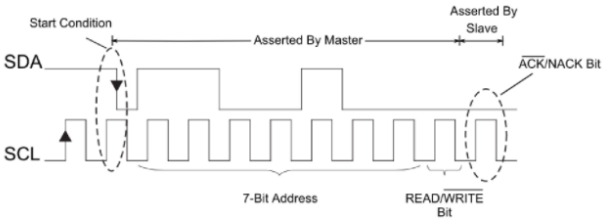
2.2. Komunikacije

Kako bi podaci mogli da se sa senzora šalju mikrokontroleru, pa onda dalje računaru, potrebno je napraviti odgovarajući vid komunikacije između uređaja. To znači da se u zavisnosti od zahteva uređaja i dostupnosti podataka bira odgovarajuća vrsta komunikacije. Projekat u sebi ima nekoliko vidova komunikacije i mogućnost zamene za neki drugi vid ukoliko korisnik to želi.

1-wire (One-wire) vrsta komunikacije koristi jednu liniju za komuniciranje. Jedan uređaj može biti master, a slave mogu biti jedan ili više uređaja. Komunikacije se odvija samo u jednom smeru istovremeno (half duplex), a brzina prenosa standarnih 15 kbps ili 111 kbps u specijalnim slučajevima. Što se tiče komunikacije sa mikrokontrolerom, 1-wire ima četiri operacije: Reset, Write 0 (Upis 0), Write 1 (Upis 1) i Read (Čitanje).

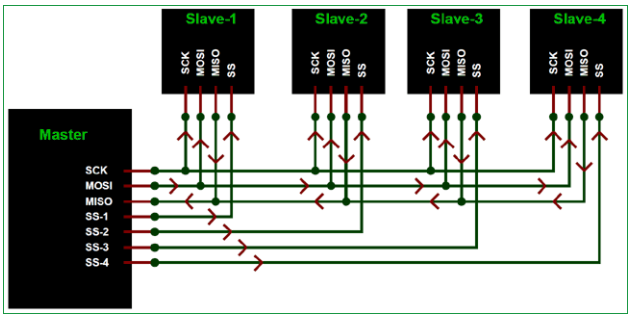
I2C (Inter-Integrated Circuit) je komunikacioni protokol dizajniran od strane kompanije Philiphs semiconductors 1980-tih godina kako bi olakšao komunikaciju između centralnog procesora i čipova na periferiji. Služi za komunikaciju na maloj razdaljini i koristi dve linije za komunikaciju u oba smera, SDA (Serial Data line) i SCL (Serial clock line). Na linije mogu biti zakačeni više master i više slave uređaja, komunikacja je sinhrona i zbog toga što koristi taktnu liniju može da ostvari velike brzine prenosa. Na slici 3. prikazan je princip rada I2C protokola.

SPI (Serial Peripheral Interface) je komunikacioni protokol namenjen za komuniciranje uređaja na maloj razdaljini. Koristi četiri linije za komunikaciju, što ga čini full-duplex komunikacijom, omogućavajući komunikaciju između master i slave uređaja istovremeno. Slave uređaji ne mogu direktno da komuniciraju sa drugim slave uređajima, odnosno komunikacija mora biti između master i slave uređaja.



SL 3. *Princip rada I2C komunikacionog protokola*

Linije potrebne za komunikaciju su SCK (Serial Clock), MOSI (Master Out Slave In), MISO (Master In Slave Out) i SS (Slave Select). Slika 4. prikazuje primer SPI komunikcije jednog master i tri slave uređaja.



SL 4. *SPI komunikacija između master i slave uređaja*

SPI i I2C komunikacije imaju svoje prednosti i mane, zbog toga se koristi onaj protokol koji više odgovara aplikaciji ili projektu. I2C je adekvatniji ukoliko postoji mnogo slave uređaja i koristi samo dve linije za komunikaciju, što ga čini konstruktivno jeftinijim rešenjem, dok SPI koristi četiri linije i za razliku od I2C, komunikacija može da budu istovremeno između master i slave i slave i master uređaja.

3. SOFTVER

U ovom odeljku opisan je FreeRTOS, njegove prednosti i neke od njegovih primena u projektu.

FreeRTOS (Free Real Time Operating System) je API koji za ulogu ima pokretanje više aplikacija istovremeno tako što ih deli u thread-ove, što je odlika današnjih operativnih sistema po čemu je dobio ime. Podrazumeva se da procesor može da vrši jednu radnju u datom trenutku, ali ukoliko postoji scheduler koji putem redosleda i prioriteta izvršavanja daje znak procesoru koju radnju treba da izvršava, smanjuje vreme koje je potrebno za izvršavanje više aplikacija. To daje prividan osećaj da se više aplikacija vrši istovremeno, pa se zbog toga kaže da on radi u realnom vremenu.

Kada neki uređaj ili sistem radi u realnom vremenu, tom sistemu se poznaje vremenski insterval u kom će u se radnja izvršiti. Sistemi koji rade u realnom vremenu mogu se podeliti u tri grupe:

1. **Soft Real Time** sisteme kod kojih radnja može da izađe iz zadatog vremenskog interval, ali tako da sistem radi funkciju za koju je namenjen.

2. **Hard Real Time** sisteme kod kojih radnja mora da se obavi u datom vremenskom intervalu.

3. **Soft i Hard Real Time** sistemi koji imaju odlike prve i druge grupe u zavisnosti koliko je sistem ili deo sistema siguran. U ovu grupu spadaju i embeded sistemi.

3.1. Task (zadatak) funkcija

Ova funkcija predstavlja osnovnu funkciju za izvršavanje neke radnje, nema povratnu vrednost i najčešće se izvršava u beskonačnoj petlji, sa mogućnošću brisanja zadatka ukoliko više nije potreban. Takođe, jedan zadatak može biti iskorišćen za kreiranje više resursno nezavisnih zadataka koji se odvojeno izvršavaju.

Aplikacije u najčešćem broju slučajeva koriste više zadataka koji rade istovremeno, tzv. multitasking. Problem kod sistema koje koriste jedno jezgro, kakav je u ovom projektu, je to što se samo jedna radnja može izvršavati u datom trenutku.

Kako bi se ovaj problem ublažio uvode se dva stanja za svaki zadatak, stanje kad se on izvršava i stanje kada čeka da bude izvršen. Pored uvođenja stanja dodaje se i FreeRTOS Scheduler, odnosno raspoređivač koji ima ulogu da odabere kada će se koji zadatak izvšavati.

Sledeći problem nastaje ako je neki zadatak bitniji od drugog. Rešenje tog problema je parametar pri kreiranju task funkcije, odnosno parametar zadužen za podešavanje prioriteta nekog zadatka.

Ukoliko je prioritet nekog zadatka veći, taj zadatak može da zabrani izvršavanje zadatka manjeg prioriteta dok se prvi zadatak ne završi ili da prekine izvršavanje zadatka sa manjim prioritetom ako je prvi zadatak bitniji od drugog.

Scheduler će uvek birati zadatak najvišeg prioriteta nakon svakog generisanog prekida (interupt).

U stanju čekanja zadatak se može nalaziti u jednom od tri stanja, odnosno blocked (blokiranom), suspended (zabranjenom) ili u ready (spremno) stanju.

Blokirano stanje predstavlja stanje u kome se nalazi zadatak dok čeka neki događaj da se desi nakon koga se zadatka izvršava. Događaji se mogu dešavati u jasno definisanom vremenskom intervalu ili nakon što im prethodi drugi događaj.

Ovim stanjem se rešava problem beskonačnog izvršavanja zadatka sa najvišim priotetom, odnosno dok je zadatak sa višim prioritetom u ovom stanju mogu da se izvršavaju zadaci nižeg prioriteta.

Zabranjeno stanje je stanje kada je zadatku onemogućen pristup za Scheduler i samim tim zadatak se ne može izvršavati. Zadatak se prebacuje u ovom stanje pomoću funkcije vTaskSuspend(), a iz stanja se izbacuje pozivom vTaskResume() ili vTaskResumeFromISR() funkcije. Često se izbegava upotreba ovog stanja u aplikacijama.

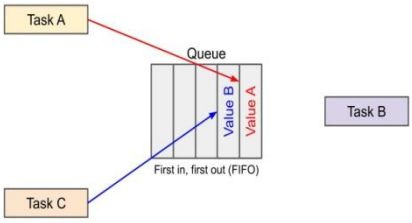
Zadatak je u spremnom stanju ako nije niti u blokiranom niti u zabranjenom stanju i čeka poziv od Schedulera kako bi se izvršio. Kada je zadatak izabran, prebacuje se u stanje izvršavanja.

Ukoliko se zadatak ne nalazi u stanju izvršavanja ili u stanju čekanja onda se sistem nalazi u stanju pripravnosti (Idle state). Stanje pripravnosti je stanje kada procesor nema aktivnih zadataka za izvršavanje, stoga on izvršava tzv. Idle Task.

To je zadatak najmanjeg prioriteta i u taj zadatak korisnik može da pozove funkciju Idle Task Hook koja može da se poziva periodično.

3.2. Queues (Redovi)

Redovi predstavljavju bazicni mehanizam komunikacije i sinhronizacije zadataka. S obzirom da nisu vezani ni za jedan zadatak, imaju mogućnost da dobijaju podatke iz više redova istovremeno, što važi i za slanje podataka u druge redove. Ukoliko ne postoje potpuni podaci za čitanje, zadatak zadužen za čitanje podataka se nalazi u blokiranom stanju, a kada pristignu podaci, automatski se zadatak prebacuje u spremno stanje. Takođe ukoliko je posle isteka vremena, koje postavlja korisnik, red prazan zadatak prelazi u spremno stanje. Na slici 5. Prikazan je primer red u kome dva zadatka upisuju, a jedan zadatak čita podatke.



SL 5. *Primer reda*

Ako aplikacija ima više zadataka koji čitaju iz reda, samo će jedan zadatak moći da pročita podatak u datom trenutku i to je zadatak sa najvećim prioritetom, a ako više zadataka imaju isti prioritet tada podatke čita zadatak koji je duže čekao u redu.

Kada zadatak koji upisuje podatke pokuša da upiše u red koji je pun, prebacuje se u blokirano stanju dok se ne oslobodi mesto u redu. Čim se mesto u redu oslobodi, zadatak prelazi u spremno stanje. Problem koji se nameće je šta ako postoji aplikacija u kojoj više zadataka upisuje u jedan red, odnosno kako da znamo koji je zadatak poslao koji segment u tom redu. Rešenje problema je kreiranje struktura podataka koje će da sadrži podatke i informacije o zadatku koji je te podatke poslao.

Kada je potrebno da se zaustavi neki zadatak koji je u stanju izvršavanja, koristi se funkcija vTaskYield(), koja prebacuje zadatak u blokirano stanje i Scheduler na njegovo mesto postavlja zadatak sa isti ili zadatak višeg prioriteta, ali ako ne postoji takav zadatak, tj. ostali zadaci su manjeg prioriteta, prvi zadatak se ponovo izvršava.

3.3. Semaphores (Semafori)

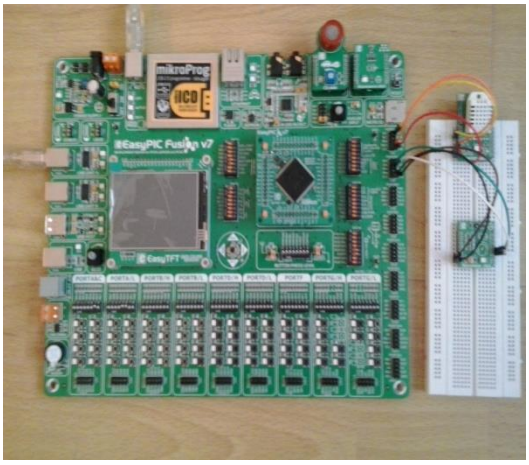
Semafori se uglavnom koriste događaji koji se dešavaju nakon prekida. Imaju dve uloge: giving (davanje) i taking (uzimanje). Uloga uzimanja je da postavi zadatak u blokirano stanje dok se ne desi određeni događaj, a uloga davanja je da nakon prekida prebaci zadatak iz blokiranog u spremno stanje. Takođe postoji i Handler task koji u zavisnosti od uloge semafora postavlja zadatak u adekvatno stanje. Semafori se dele na binarne i brojačke.

Binarni semafori predstavljaju red koji ima jedan element. Ukoliko je red prazan i zadatak pokuša da čita iz tog reda, zadatak će biti poslan u blokirano stanje. Nakon što se desi prekid i generisana je funkcija davanja, red dobija token i tada je popunjen. Zatim se izvršava Handler task koji zadatka koji treba da čita iz reda šalje u spremno stanje. Kada zadatak pročita iz reda, ponovo se šalje u blokirano stanje. Problem kod ovakvog načina rada apli-

kacije jeste to što se prekid može izvšiti samo jednom tokom izvršavanja zadatka, a ako se desi da više prekida, tada se gube događaji.
Rešenje problema je upotreba brojačkog semafora. Za razliku od binarnog semafora koji se posmatra kao red sa jednim elementom, brojački semafor se posmatra kao red sa više istih. Oni se koriste u dva slučaja: za brojanje događaja i za upravljanje resursima. Za brojanje događaja se koriste tako što svaki prekid generiše funkciju davanja koja dodaje element u red, dok se funkcijom uzimanja, odnosno čitanjem iz reda, smanjuje broj elemenata u redu. Uloga upravljanja resursima je takva da broj elemenata u redu predstavlja broj raspoloživih resursa, kada zadatak uzme element iz reda smanjuje se broj resursa, dok se red ne isprazni, a kada zadatak završi sa resursom vraća ga na kraj reda [5].

5. TESTIRANJE SISTEMA

Za početak je potrebna kalibracija senzorskih modula i njihovo povezivanje na razvojnu ploču EasyPIC Fusion v7. Senzorski moduli su povezani sa pločom putem mikroBUS™ konektora, ali pošto na ploči postoje dva takva konektora, druga dva senzora su povezana direktno preko pinova. Nakon spajanja senzorskih modula, povezaje se razvojna ploča sa računarom putem UART konektora, odnosno USB priključa na računaru.
Kako bi senzorski moduli slali podatke, potrebno je spustiti kod za svaki od senzora na ploču i na kraju je potrebno napisati odgovarajući kod za hardverski deo aplikacije. Taj kod se sastoji od postavljanja zadataka za svaki senzor pojedinačno i zadatka za slanje podataka računaru. Postavljeni su odgovarajući prioriteti na zadatke kako ne bi došlo do gubljenja podataka i postavljeni su tajmeri za svaki od senzora. Ovaj deo koda napisan je u jeziku MikroC za PIC32 kompanija MikroElekronika. Izgled povezane ploče sa senzorima prikazan je na slici 6.



SL 6. Razvojna ploča sa senzorkim modulima

Kada podaci stignu na računar potrebno ih je parsirati i smestiti u bazu podataka. Za ovaj deo projekta korišćen je programski jezik Python. Kreirane su dve aplikacije, jedna koja ima ulogu da primi podatke i parsira ih, dok druga pokreće bazu podataka u upisuje podatke u nju.

Prva aplikacija posmatra da li mikrokontroler šalje podatke i ako je to slučaj razdvaja ih u zavisnosti od kog su senzorskog modula došli i dodaje je im vreme i datum po kome će podaci biti sortirani. Ova aplikacija provera da li su pristigli podaci svake sekunde i nakon što ih parsira, pravi rečnik u kome upisuje vrednosti i raspoređuje ih na odgovarajući način.
Druga aplikacija koristi MongoDB bazu podataka, ona proverava da li su podaci odgovarajućeg formata i upisuje ih u bazu podataka. Za preuzimanje podataka iz baze i pokretanje veb stranice korišćeni su JavaScript, HTML i CSS. Podaci se uzimaju iz baze podataka, prave se odgovarajući grafici i tabele, određuju se minimalne, maksimalne i srednje vrednosti koje se predstavljaju na veb stranici.

6. ZAKLJUČAK

Opisani projekat uz male modifikacije može da se koristi i u komercijalne svrhe, trenutno je namenjen za kućnu upotrebu, takođe projekat nije ograničen na senzorske module koji su korišćeni u njemu. Dakle, moguće je korišćenje drugih senzorskih modula kao i merenje drugih veličina uz odgovarajuću opremu..
Prednosti ovakvog sistema su njegova jednostavnost i prikaz podataka u realnom vremenu. Sistem je napravljen da radi u lokalnoj mreži, ali postoji mogućnost povezivanja na gloablnu internet mrežu. U tom slučaju korisnik može lakše da pristupi podacima dobijenim od senzora, ali tada bi mogla da se pojavi potrencijalna pretnja korišćenja podataka od strane trećeg lica koje ne bi trebalo da ima pristup podacima. Ukoliko bi korisnik želeo, sistem je moguće napajati baterijskim izvorom napajanja. Tako bi on dobio na mobilnosti, ali baterije nisu namenjene da traju duži vremenski period, odnosno stalna kupovina baterija bi bila dodatni trošak.

Kao poboljšanje, sistemu se može dodati aplikacija ili deo aplikacije koji prikazuje podatke iz nekog vremenskog perioda, kao grafički interfejs za lakši pristup podacima. Sistemu se takođe može dodati aplikacija koja posmatra da li je veličina u dobrom opsegu i prijavi korisniku da je neka od veličina van opsega ili da primeni odgovarajuće mere ukoliko korisnik to dozvoli.

7. LITERATURA

[1] https://www.mikroe.com/dht22-click
[2] https://www.mikroe.com/barometer-click
[3] https://www.mikroe.com/co-click
[4] https://www.mikroe.com/illuminance-click
[5] Materijali iz predmeta Merenja u realnom vremenu

Kratka biografija:

Stefan Tešanović rođen je 1996. godine u Novom Sadu. Diplomirao je na fakultetu tehničkih nauka u Novom Sadu na katedri za električna merenja 2019. godine.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.38
DOI: https://doi.org/10.24867/15BE19Ilic

PROGRAMABILNO NAPAJANJE NAMENJENO UREĐAJIMA U EKSPLOZIVNIM ZONAMA

PROGRAMMABLE POWER SUPPLY FOR DEVICES OPERATING IN EXPLOSIVE ZONES

Damir Popov, Vladimir Rajs, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom projektu opisan je način izrade uređaja koji će omogućiti napajanje potrošača u eksplozivnoj zoni, a koji će istovremeno omogućiti korisniku podešavanje potrebnog napona potrošača putem jednostavne aplikacije. Prikazani su rezultati simulacija i rezultati merenja, a sve korišćene komponente detaljno su opisane.*

Ključne reči: *Digitalni potenciometar, strujna povratna sprega, svojstvena bezbednost, Zener barijera.*

Abstract – *This project describes the manufacturing process of a device that will be able to power consumers in explosive zones, that will also at the same time enable the user to set the necessary voltage using a simple application. The results of simulations and measurements are shown, and a detailed description of used components is included.*

Keywords: *Digital potentiometer, current feedback, intrinsically safe, Zener barrier.*

1. UVOD

Uvođenjem električne opreme u industriju i domaćinstva, a potom metana i ugljene prašine, čoveku je bilo je neophodno garantovati sigurnost i bezbenost. Iako su uložena velika finansijska sredstva i napori za razvoj protiveksplozivne zaštite, dobit koja proizilazi od električnih uređaja je veća. To je bio podstrek za neprekidnim razvojem i usavršavanjem protiveksplozivne opreme.
Danas, nakon loših iskustava i podnetih velikih žrtava, može se smatrati da je protiveksplozivna zaštita dostigla zavidan nivo. Propisani su standardi koje uređaji namenjeni za rad u eksplozivnim sredinama moraju da ispoštuju, a sve to u cilju što bezbednijih sistema i kvalitetnije protiveksplozivne zaštite.

2. ANALIZA PROBLEMA

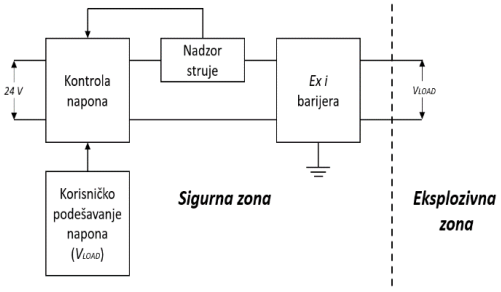
Sa početkom korišćenja prirodnog gasa u domaćinstvima i industriji, javila se potreba za nadzorom njegove potrošnje. Za precizno merenje potrošnje gasa koriste se korektrori zapremine gasa. Oni, na osnovu temperature i pritiska gasa, kao i na osnovu informacija dobijenih sa gasnog brojila, precizno proračunavaju protok gasa. Većina korektora zapremine ima baterijsko napajanje, ali kod mnogih postoji i opcija eksternog napajanja.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Rajs, vanr. prof.

Ideja je isprojektovati napajanje takvo da se njime mogu napajati različiti tipovi korektora zapremine. Naponi napajanja većine zastupljenih korektora su u opsegu od 4 V do 12 V, sa potrošnjom struje koja nije veća od 70 mA. S obzirom na činjenici da se korektori zapremine nalaze u eksplozivnim zonama, projektovano napajanje mora zadovoljiti određene kriterijume kako bi bilo prigodno za ovu namenu.
Zener barijera potpada pod uređaje svojstvene bezbednosti *„Ex i“* koji eksploziju sprečavaju ograničavajući energiju na manju od potrebne energije za paljenje eksplozivne atmosfere. Sama barijera se može nalaziti i u sigurnoj zoni, ali napon i struja koje ona isporučuje, kao i ostala pravila propisana su sa standardima *SRPS EN 60079-11 2012 [1]* i *SRPS EN IEC 60079-0 2019 [2]*. Radi jednostavnosti, ženjeni napon korektora zapremine, korisnik može da podešava softverski *[3], [4]*.

3. IDEJNO REŠENJE

Na slici 1 prikazana je blok šema projektovanog uređaja. Ulazni napon od 24 V potrebno je prilagoditi željenom naponu potrošača. Pošto se potrošač nalazi u eksplozivnoj zoni, neophodno je obezbediti da napon i struja koji ulaze u nju ne budu veći od maksimalnih dozvoljenih vrednosti.

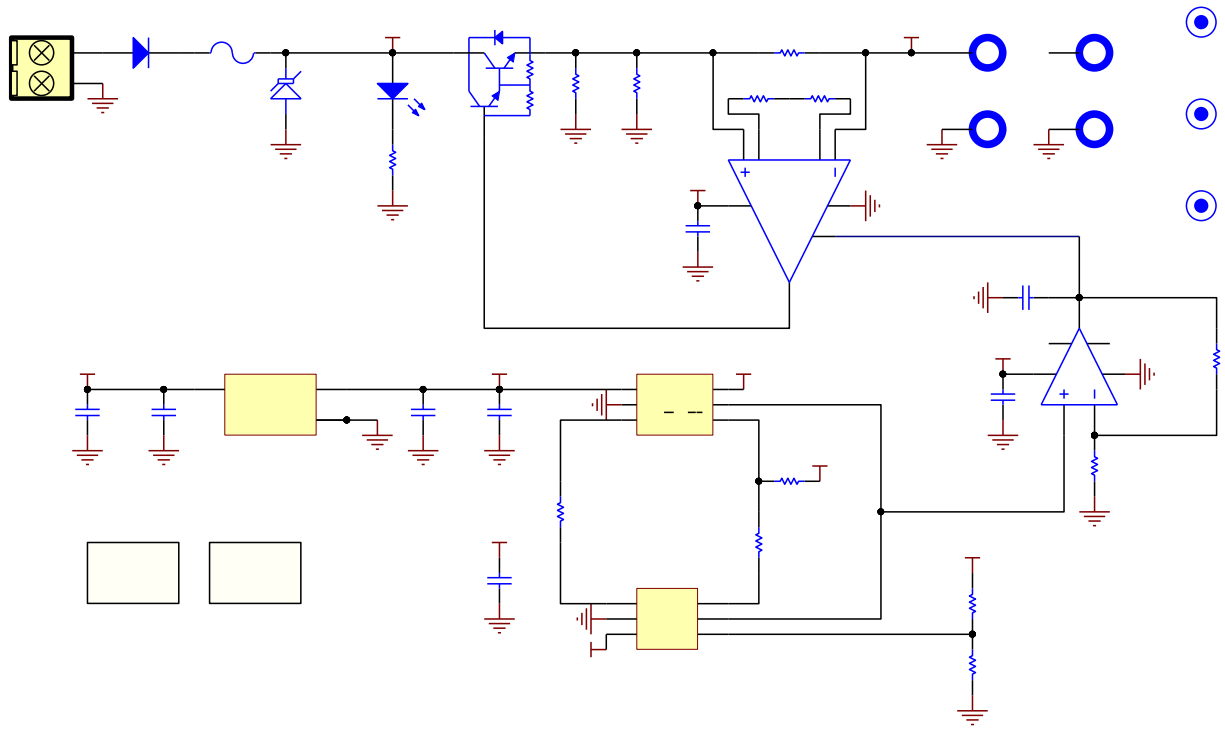


Slika 1. Blok šema uređaja

Kao ograničavač struje i napona koristi se svojstveno siguran uređaj, odnosno *Ex i* barijera, koja u kolo unosi rednu otpornost. U zavisnosti od struje potrošača, na barijeri će se javiti proporcionalan pad napona.
Ideja je da se na korisniku ostavi mogućnost softverskog podešavanja potrebnog napona potrošača. Nezavisno od toga, vrši se nadzor struje potrošača.
U blok za kontrolu napona, na osnovu izmerene struje i poznavanja otpornosti *Ex i* barijere, procenjuje se koliki će biti pad napona na barijeri. Taj napon se sabira sa korisnički zahtevanim naponom i kao takav prolazi preko *Ex i* barijere u eksplozinu zonu do potrošača.

4. PAKTIČNO REŠENJE PROGRAMABILNOG NAPAJANJA

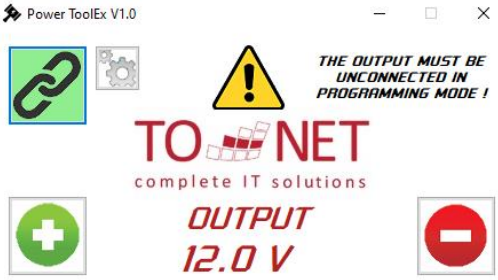
Na slici 2 prikazana je projektovana šema programabilnog napajanja. Na samom ulazu nalazi se zaštitno kolo u vidu diode D_1, *MRA4004T3G [5]*, koja štiti uređaj od inverzne polarizacije napona, osigurača F_1, *MRF 315 [6]*, kao zaštita od prekomerne struje i TVS diode D_2, *SMF30VTR [7]*, koja predstavlja ESD zaštitu. Pored zaštite postoji i svetlosna indikacija uključenosti uređaja u vidu LED, *APT1608CGCK [8]*.



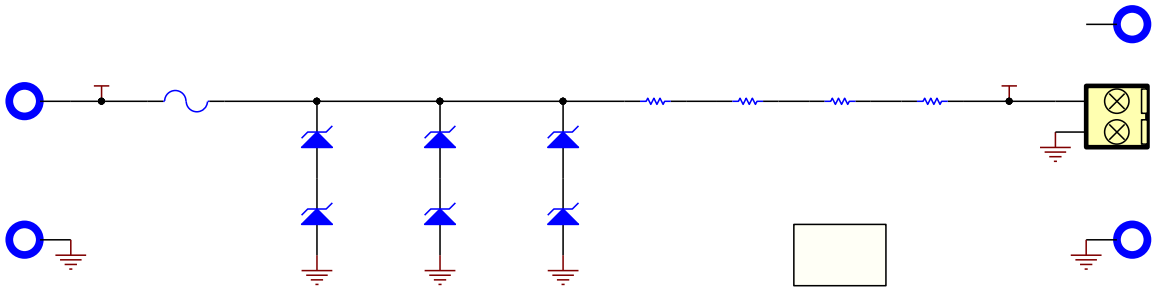
Slika 2. Šema programabilnog napajanja

Linearni stabilizator U_4, *MCP1799T-3302H/DB [9]*, spušta ulazni napon na 3,3 V i napaja digitalni potenciometar U_3, *MCP4023T-103E/CH [10]*. Pomeranjem klizača potenciometra programiranjem, na njemu će se pojaviti različit napon, u opsegu od 0 do 3,3 V, prouzrokovan promenom otpornosti u formiranom naponskom razdelniku. Bitno je naglasiti da se pozicija klizača čuva u internoj *EEPROM* memoriji, te da će i pri resetovanju uređaja (npr. usled nestanka struje) klizač zadržati posledne podešenu poziciju. Napon dobijen na klizaču potenciometra potrebno je prilagoditi željenom opsegu napona potrošača. U tu svrhu korišćen je operacioni pojačavač U_2, *TL071CDRE4 [11]*, u konfiguraciji neinvertujućeg pojačavača pojačanja 4, kojem se pojačanje podešava otpornicima R_9 i R_{13}. Izlazni napon ovog pojačavača predstavlja željeni napon potrošača uvećan za vrednost napona baza-emiter Darlington tranzistora Q_1, *MJD122G [12]*. Tako dobijeni napon dovodi se na referentni pin instrumentacionog pojačavača U_1, *INA826AIDR [13]*. Instrumentacionim pojačavačem je omogućena strujna povratna sprega, koja obezbeđuje da pad napona na otpornicima R_1, R_2, R_3 i R_4 Zener barijere (slika 4), prouzrokovan strujom potrošača, ne utiče na napon potrošača. U zavisnosti od struje potrošača na otporniku R_1 će se formirati pad napon koji će biti pojačan i predstavljaće baš onoliku vrednost napona koja će se izgubiti na otpornicima Zener barijere i samom otporniku R_1. Taj napon se sabira sa naponom dovedenim na referentni pin instrumentacionog pojačavača. Proračunato pojačanje instrumentacionog pojačavača je 10,16 i podešava se otpornicima R_4 i R_6. Instrumentacioni pojačavač nema dovoljene strujne mogućnosti da napaja potrošač, pa je iz tog njegov izlaz doveden na bazu Darlington tranzistora u konfiguraciji *emitter-folower*-a. Otpornici R_3 i R_8 imaju ulogu da drže tranzistor u aktivnom režimu i pri nultoj struji potrošača čime se obezbeđuje da je pad napona baza-emiter uvek bude približno isti.

Naponski razdelnik formiran od otpornika R_{12} i R_{14} ima ulogu da prilikom porogramiranja potenciometra korisniku bude omogućeno da vidi ulazni napon Zener barijere preko jednog analognog pina mikrokontrolera. Digitalni potenciometar se programira jednostavnim *Up/* $\overline{Down}$ protokolom, za koji su potrebna dva digitalna izlaza mikrokontrolera. U ovom projektu se kao programator koristi *Arduino Nano [14]*. Kako bi korisnik na jednostavan način mogao da podesi željeni izlazni napon potrošača, razvijena je aplikacija za programiranje digitalnog potenciometra. Izgled aplikacije za podešavanje napona potrošača, u slučaju kada je napon podešen na 12 V, prikazan je na slici 3.



Slika 3. Izgled alplikacije za podešavanje napona potrošača u slučaju kada je napon podešen na 12 V



Slika 4. Šema Zener barijere

5. PRAKTIČNO REŠENJE ZENER BARIJERE

Na slici 3 prikazana je projektovana šema Zener barijere *„Ex ia“* nivoa zaštite. Ovaj tip zaštite je razlog postojanja tri paralelne grane sa Zener diodama. Time se obezbeđuje da i pri otkazivanju dve Zener diode u dve različite grane, bilo da otkažu formirajući otvorenu ili kratku vezu, u eksplozivnu zonu ne prođe napon veći od dozvoljenog. Kada ulazni napon premaši nominalnu vrednost napona Zener grane od 20 V, Zener diode provedu, održavajući napon konstantnim, a višak struje odvode ka masi. Kako bi jedna Zener grana mogla da izdrži veću vrednost struje, korišćene su dve *1N5347BG [15]* Zener diode nominalnog napona 10 V i snage 5 W, povezane redno. Osigurač *0242.100UR* na ulaznom delu kola ima ulogu da zaštiti Zener diode ako struja kroz njih postane prevelika, tj. struja pregorevanja osigurača je manja od struje koju Zener diode mogu da izdrže. Pomenuti osigurač će pregoreti pre Zener dioda i u slučaju da je ulazni napon 250 V, a s obzirom na to da ima prekindnu struju veću od 1,5 kA, obezbeđeno je da će i u ovom slučaju kolo biti prekinutno na siguran način. Otpornici R_1, R_2, R_3 i R_4 imaju ulogu da ograniče izlaznu struju tako da i u slučaju kratkog spoja napajanja i mase ona ne premaši dozvoljenu vrednost. Maksimalna struja koja se sme pojaviti u eksplozivnoj zoni za napon od 21 V (uzimajući u obzir toleranciju Zener dioda od +5 %) je 262 mA. Razlog za korišćenje četiri umesto jednog otpornika je veća snaga koju oni mogu da izdrže. Uzimajući u obzir njihovu toleranciju od 1 %, kao i hladnu otpornost osigurača, ograničiće struju kratkog spoja na 231,45 mA, što je manje od kritične.

Bitno je napomenuti da su sve komponente i njihove vrednosti birane na osnovu uslova propisanih sertifikatom *SRPS EN 60079-11 2012* za standardizaciju. Pored toga, posebna pažnja je obraćena na crtanje štampane pločice Zener barijere, gde su pravila crtanja takođe propisana istim sertifikatom. Pločica Zener barijere zaštićena je epoksidnom smolom čime se pored smanjenih zahteva za rastojanja između komponenti i vodova prilikom crtanja štampane ploče, postiže i to da korisnik ne može da menja originalne komponente. Zener barijeru je neophodno uzemljiti. To je postignuto povezivanjem metalne pločice, namenjene za pričvršćivanje uređaja na *DIN* šinu, na masu komplertnog uređaja. U slučaju da *DIN* šina nije uzemljena, ili da se uređaj ne montira na nju, alternativno postoji mogućnost povezivanja uzemljenja preko provodnog odstojnika povezanog sa masom uređaja, a kojem korisnik ima pristup na poklopcu kutije uređaja.

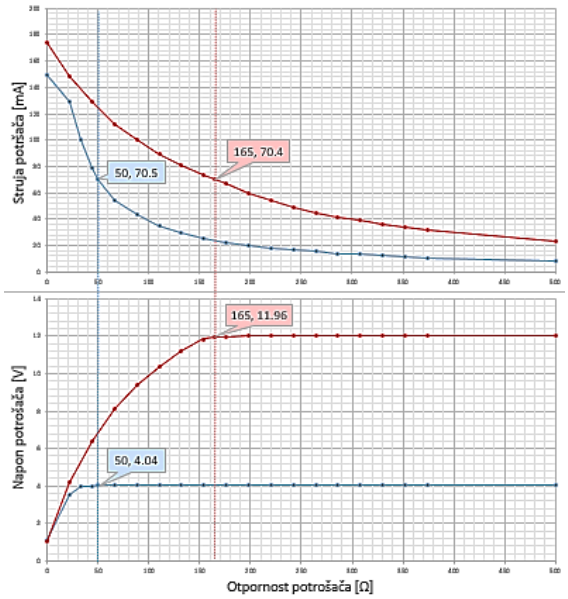
6. REZULTATI

Simulacije su vršene u programskom paketu *Micro-Cap 12*. Većina komponenti korišćenih u simulaciji su identične kao i one u praktičnoj ralizaciji. Umesto komponenti koje ne postoje u bibliotekama programskog paketa *Micro Cap 12* korišćena je adekvatna zamena sa vrlo sličnim karakteristikama. Korišćena je DC analiza sa parametrizovanom otpornošću potrošača čime se simulira kako se kolo ponaša za različite struje potoršača.

U slučaju da je podešeni napon potrošača 12 V, za struje ispod 70 mA, napon je konstantanih 12 V. Kada struja poraste iznad 70 mA, napon počinje da opada sa porastom struje. Ova pojava se dešava zbog činjenice da je napon koji instrumentacioni pojačavač treba da isporuči na svom izlazu u tom slučaju veći od maksimalnog napona koji on može da da. Važno je napomenuti da ulazni napon Zener barijere ne bi trebalo da bude veći od 19 V jer bi postojala mogućnost da Zener diode provedu i napon potrošača više ne bi bio konstantan. Pored toga, usled prevelikog ulaznog napona, kada zbog provodnih Zener dioda, struja osigrača postane prevelika i on pregori, kolo bi prestalo da radi. U slučaju da je podešeni napon potrošača 4 V, uređaj bi mogao da ispravno radi i za veće struje potrošača. Međutim, struja potrošača je ograničena i osiguračem Zener barijere, stoga ona ne može biti ni u kom slučaju veća od 100 mA.

Rezultati simulacija potvrđeni su i testiranjem uređaja u realnim uslovima. Za potrebe testiranja, razvijena je štampana pločica koja simulira potrošač. Dobijeni grafici napona u zavisnosti od struje potrošača, prikazani na slici 5, su u skladu sa onim dobijenim simulacijom uz minimalne razlike prouzrokovane neidealnim mernim instrumentima, činjenicom da nisu sve komponente u simulaciji identične sa onim u praktičnoj realizaciji, kao i većim korakom promene otpornosti potrošača tj. promenom struje potrošača.

Potrebno je naglasiti, da će napon potrošača ipak u određenoj meri varirati zbog zagrevanja komponenti. Naime, grejanjem otpornici gube na otpornosti. Komponenta koja najvise disipira toplotu je Darlington tranzistor kojem sa porastom struje potrošača raste i snaga disipacije, a sa porastom napona potrošača, disipacija opada. U najgorem slučaju, kada je podešen napon potrošača od 4 V i kada je struja 70 mA snaga disipacije Darlington tranzistora biće 1,3 W i tranzistor će dostići temperaturu od oko 73 ºC.

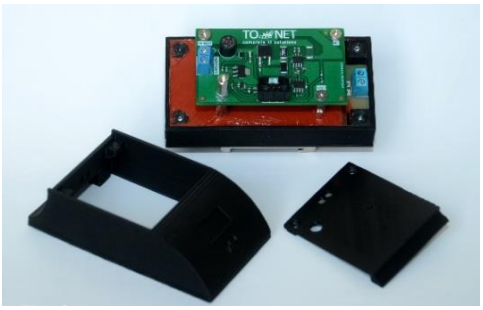


Slika 5. *Grafici merene struje (gornji grafik crveno) i merenog napona (donji grafik crveno) potrošača u slučaju da je željeni napon potrošača 12 V i grafici merene struje (gornji grafik plavo) i merenog napona (donji grafik plavo) potrošača u slučaju da je željeni napon potrošača 4 V*

Posle 4 sata rada uređaja u ovakvom režimu napon potrošača, umesto željenih 3,95 V biće 4,23 V. Dodatno, ako struja potrošača tada naglo padne na 0, napon će skočiti na 4,58 V. Ovo se dešava zbog činjenice da se pločice programabilnog napajanja i Zener barijere ne hlade na isti način. Iz istog razloga, zbog zagrevanja, u trenutku uključivanja uređaja, postojaće blagi pad napona na potrošaču dok se temperature pločica ne izjednače. Merenja su vršena pri sobnoj temperaturi od 25 ºC. Pri drugim okolnim temperaturama moguća su drugačija odstupanja.

7. ZAKLJUČAK

Ovim radom je opisana moguća implementacija programabilnog napajanja namenjenog za uređaje u potencijalno eksplozivnim sredinama. Predlog za eliminaciju zavisnosti napona potrošača od njegove struje, nalazi se u strujnoj povratnoj sprezi programbilnog napajanja. Prikazane su neke od mogućih rešenja protiveksplozivne zaštite, dok je svojstveno sigurna zaštita Zener barijerom u ovom uređaju primenjena i detaljno analizirana. Na slici 6 predstavljen je finalni izgled uređaja.



Slika 6. *Finalni izgled uređaja*

8. LITERATURA

[1] Institut za standardizaciju Srbije, *Eksplozivne atmosfere – Deo 11: Oprema zaštićena svojstvenom bezbednošću "i", SRPS EN 60079-11*, Beograd, novembar 2012.

[2] Institut za standardizaciju Srbije, *Eksplozivne atmosfere – Deo 0: Oprema – Opšti zahtevi, SRPS EN IEC 60079-0*, Beograd, jul 2019.

[3] Radovan M. Jovanov, *Protiveksplozivna zaštita električnih i neelektričnih uređaja i instalacija*, Institut za nuklearne nauke „Vinča", Beograd, mart 2015.

[4] Hans-Jürgen Linström, Johannes Buhn, Karel Neleman, *Basic concepts for explosion protection*, dostupno na: *https://www.bartec.de/en/downloads/safety-academy/ex-protection.pdf*, datum pristupa: 25.8.2021.

[5] Onsemi, Datasheet diode MRA4004T3G, dostupno na: *https://www.onsemi.com/pdf/datasheet/mra4003t3-d.pdf*, datum pristupa: 25.08.2021.

[6] Bel Fuse, Datasheet osigurača MRF 315, dostupno na: *https://www.belfuse.com/resources/datasheets/circuitprotection/ds-cp-mrf-series.pdf*, datum pristupa: 25.08.2021.

[7] ROHM Semiconductor, Datasheet TVS diode SMF30VTR, dostupno na: *https://rohmfs-rohm-com-cn.oss-cn-shanghai.aliyuncs.com/en/products/databook/datasheet/discrete/diode/zener/smf30vtr-e.pdf*, datum pristupa: 25.08.2021.

[8] Kingbright, Datasheet LED APT1608CGCK, dostupno na: *https://www.kingbrightusa.com/images/catalog/SPEC/APT1608CGCK.pdf*, datum pristupa: 25.08.2021.

[9] Microchip Technology , Datasheet linearnog stabilizatora napona MCP1799T-3302H/DB, dostupno na: *https://ww1.-microchip.com/downloads/en/DeviceDoc/MCP1799-Data-Sheet-20006248A.pdf*, datum pristupa: 25.08.2021.

[10] Microchip Technology , Datasheet digitalnog potenciometra MCP4023T-103E/CH, dostupno na: *https://ww1.-microchip.com/downloads/en/DeviceDoc/21945e.pdf*, datum pristupa: 25.08.2021.

[11] Texas Instruments, Datasheet operacionog pojačavača TL071CDRE4, dostupno na: *https://www.ti.com/lit/ds/symlink-/tl071.pdf?ts=1629703416751&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FTL071*, datum pristupa: 25.08.2021.

[12] Onsemi, Datasheet Darlington tranzistora MJD122G, dostupno na: *https://www.onsemi.com/pdf/datasheet/mjd122-d.pdf*, datum pristupa: 25.08.2021.

[13] Texas Instruments, Datasheet instrumentacionog pojačavača INA826AIDR, dostupno na: *https://www.ti-.com/lit/ds/symlink/ina826.pdf?ts=1629717211627&ref_url=https%253A%252F%252Fwww.google.com%252F*, datum pristupa: 25.08.2021.

[14] Arduino, Datasheet razvojnog okruženja Arduino Nano, dostupno na: *https://www.arduino.cc/en/uploads/Main/-ArduinoNanoManual23.pdf*, datum pristupa: 25.08.2021.

[15] Onsemi, Datasheet diode 1N5347BG, dostupno na*:* *https://www.onsemi.com/pdf/datasheet/1n5333b-d.pdf*, datum pristupa: 25.08.2021.

Kratka biografija:

Damir Popov rođen je u Vršcu 1996. godine. Osnovne akademske studije završio na Fakultetu tehničkih nauka 2019. godine, oblast Elektrotehnika i računarstvo, smer Primenjena elektronika. Master rad na istom fakultetu, oblast Elektrotehnika i računarstvo, smer Primenjena elektronika, odbranio je 2021. god.

Zahvalnica:

Ovaj rad je podržan od strane Fakulteta tehničkih nauka u Novom Sadu, Departman za energetiku elektroniku i telekomunikacije, u okviru realizacije projekta pod nazivom: „Istraživanja u oblasti energetike, elektronike, telekomunikacija i primenjenih informacionih sistema u cilju modernizacije studijskih programa".

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/15BE36Skrbic

SLOJ ZA PERZISTENCIJU PODATAKA U ČISTO FUNKCIONALNOJ PROGRAMSKOJ PARADIGMI

DATA PERSISTENCE LAYER IN A PURELY FUNCTIONAL PROGRAMMING PARADIGM

Milan Škrbić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je opisana teorija kategorija i njena primena u Haskell programskom jeziku. Pored teorijskog dela prikazana je i konkretna implementacija sloja za perzistenciju prodataka u Instagram klon aplikaciji. Aplikacija je implementirana u IHP radnom okviru.*

Ključne reči: *Funkcionalno programiranje, funkcija, funktor, monada, teorija kategorija, aplikativni funktor, produkt, koprodukt, transformacija*

Abstract – *The paper describes the category theory and its usage in the Haskell programming language. In addition to the theoretical part, a concrete implementation of the layer for data persistence in the Instagram clone application is shown. The application is implemented in the IHP framework.*

Keywords: *Functional programming, function, functor, monad, category theory, applicative functor, product, coproduct, transformation*

1. UVOD

U ovom radu prikazana je primena koncepata iz teorije kategorija u programskom jeziku Haskell, kao i implementacija aplikacije Instagram klona u IHP-u (Integrated Haskell Platform). Implementacija aplikacije se koristi radi prikazivanja primene koncepata teorije kategorija u Haskell programskom jeziku.

2. FUNKCIONALNA PROGRAMSKA PARADIGMA

Jedan od većih problema objektno orijentisanog programiranja jeste to što svaka funkcija može da ima bočne efekte koje je teško pratiti u složenijim sistemima i često se dešava da baš ovaj problem proizvede veliki broj neočekivanih ponašanja u toku izvršavanja programa.

2.1 Razlike funkcionalnog i objektno orijentisanog programiranja

U funkcionalnom programiranju, kako mu ime i sugeriše, funkcija igra veliku ulogu. Za razliku od objektno orijentisanog programiranja, u funkcionalnom programiranju svaka funkcija mora da bude čista (engl. pure function), ili drugačije rečeno matematička funkcija.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Milan Segedinac, vanr. prof.

Objektno orijentisani jezici su dobri kada je prisutan fiksni skup operacija na stvarima, a kako se kod razvija, primarno se dodaju nove stvari. To se može postići dodavanjem novih klasa koje primenjuju postojeće metode, a postojeće klase ostaju nepromenjene.
Funkcionalni jezici su dobri kada je prisutan fiksni skup stvari, a kako se kod razvija, primarno se dodaju nove operacije na postojećim stvarima. To se može postići dodavanjem novih funkcija koje računaju sa postojećim tipovima podataka, a postojeće funkcije ostaju nepromenjene [3].

3. TEORIJA KATEGORIJA I HASKELL

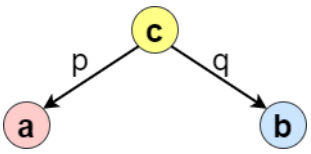
Kategorija se sastoji od objekata i strelica između tih objekata. Objekat u kategoriji je apstraktna stvar i može da se odnosi na razne pojmove. Strelice u kategoriji se drugačije nazivaju morfizmima. Konkretan primer jedne kategorije je kategorija skupova. U kategoriji skupova objekti su skupovi, dok su morfizmi funkcije koje mapiraju elemente jednog skupa na elemente drugog skupa.
Svaka kategorija mora da poštuje dva pravila, pravilo kompozicije i pravilo identiteta. Kompozicija se odnosi na to da se morfizmi spajaju tako da ako postoji strelica *f* od objekta *A* do objekta *B* i još jedna strelica *g* od objekta *B* do objekta *C*, onda mora postojati strelica, njihova kompozicija, koja ide od *A* do *C* koja se označava sa $g \circ f$ i čita se kao *„g posle f“*.
Što se tiče identiteta, u kategoriji svaki objekat mora da ima morfizam koji spaja taj objekat sa samim sobom. To znači da kada je identitet spojen sa bilo kojim drugim morfizmom koji započinje iz tog objekta, ili se završava u tom objektu, daje isti morfizam. Morfizam identiteta za objekat *A* se naziva **id**$_A$.

2.2. Tipovi i Funkcije

Funkcije u Haskellu imaju svoje tipove. Svaka funkcija se deklariše tako što se navode tipovi parametara te funkcije i odvojeni su strelicama. Poslednji tip naveden u deklaraciji funkcije ne predstavlja parametar nego povratnu vrednost funkcije. Funkcija čija deklaracija je *Integer -> Integer -> Bool*, prima 2 parametra tipa *Integer* i vraća povratnu vrednost tipa *Bool*.

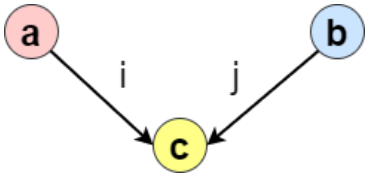
2.3. Produkti i koprodukti

Na slici 1 objekat *c* predstavlja proizvod objekata *a* i *b*. Definitivno je da ima još objekata, *c1, c2, c3...* koji imaju morfizme ka objektima *a* i *b*. Kako da znamo da je baš *c* produkt objekata *a* i *b*? Da bi *c* bio produkt objekata *a* i *b* mora da postoji unikatni morfizam *m* od svih ostalih „lažnih produkata“ ka objektu *c*.



Slika 1. *Produkt [1]*

Na slici 2 vidimo da je koprodukt dva objekta a i b objekat c sa dve injekcije takve da za bilo koji drugi objekat c1,c2,c3... sa dve injekcije postoji jedinstveni morfizam m od c do „lažnog koprodukta“ koji faktorizuje te injekcije.



Slika 2. *Koprodukt [1]*

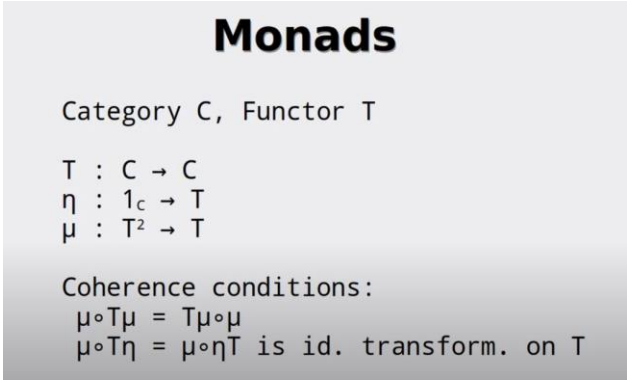
2.4. Funktori

Kada pričamo o **funktorima** želimo da podignemo nivo apstrakcije. Sada više ne gledamo morfizme između objekata u kategoriji, već gledamo morfizme između samih kategorija. Funktori predstavljaju baš to, morfizme između kategorija. Međutim, funktori ne preslikavaju samo objekte jedne kategorije u objekte druge kategorije, nego preslikavaju i morfizme između tih objekata.

2.5. Prirodne Transformacije

Prirodne transformacije (*engl. Natural transformations*) su mapiranja funktora - posebna mapiranja koja čuvaju svoju funkcionalnu prirodu.

2.6. Monade



Slika 3. *Definicija Monade [2]*

Na slici 3 možemo da vidimo konkretnu matematičku definiciju **monade**. Monadu predstavlja kategorija **C** i funktor *T*, koji mapira kategoriju **C** u samu sebe. Funktor koji mapira kategoriju u samu sebe se naziva **endofunktor**. Takođe, monadu čine i dve prirodne transformacije – μ i η, koje su veoma slične prirodnim transformacijama kod monoida. Transformacija η mapira funktor identiteta u funktor *T*, dok μ mapira dve aplikacije funktora *T* u jedan funktor *T*.

Monade kontrolišu lančanja funkcija i to podseća na imperativno programiranje. Međutim ovo nije samo još jedan način imperativnog programiranja, već prava snaga monada se ogleda u tome što nam omogućavaju kontrolisanu kompoziciju funkcija i na taj način rešavaju veliki problem funkcionalnog programiranja – kako rukovanja bočnim efektima.

3. PRIMENA KONCEPATA TEORIJE KATEGORIJA NA PRIMERU INSTAGRAM KLON APLIKACIJE

U ovom poglavlju prikazani su primeri korišćenja koncepata iz teorije kategorija u Haskellu na praktičnom primeru Instagram klon aplikacije, koja je napisana u IHP (Integrated Haskell Platform) radnom okviru. Naziv aplikacije je *„Polaroid“*.

```
39  data User' comments followsFollowers followsUsers likes posts =
40      User {id :: (Id' "users"),
41            email :: Text,
42            passwordHash :: Text,
43            lockedAt :: (Maybe UTCTime),
44            failedLoginAttempts :: Int,
45            firstName :: Text,
46            lastName :: Text,
47            pictureUrl :: (Maybe Text),
48            comments :: comments,
49            followsFollowers :: followsFollowers,
50            followsUsers :: followsUsers,
51            likes :: likes,
52            posts :: posts,
53            meta :: MetaBag
54      } deriving (Eq, Show)
```

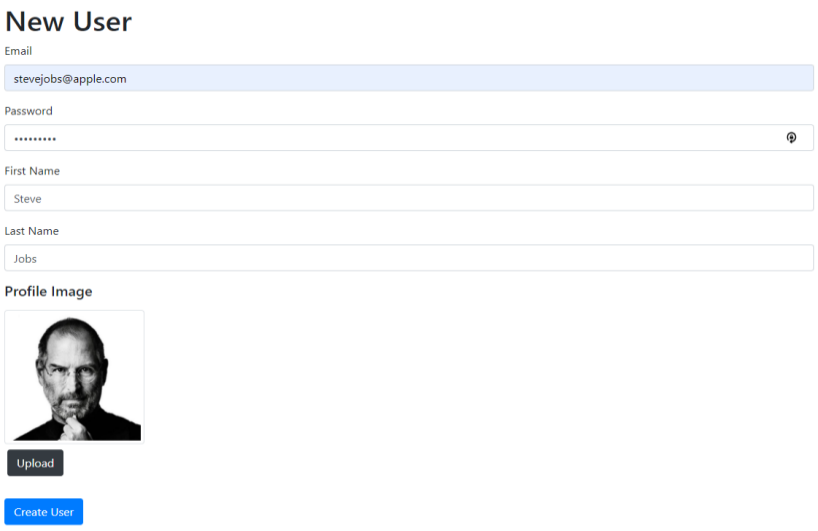
Slika 4. *User tip podataka*

Na slici 4 je dat praktičan primer *User* tipa podataka.
Sloj zadužen za upravljanje i perzistenciju podataka se nalazi u folderu *Controller*.
Na primeru aplikacije veliki broj funkcija je spojen korišćenjem **monada**, tj. „**do**“ notacije i operatora kao što je **(>>=)**.
WelcomeAction je funkcija koja, u zavisnosti od toga da li postoji ulogovani korisnik ili ne, izvlači iz baze podataka preko složenog SQL upita sve objave koje pripadaju svim korisnicima koje ulogovani korisnik prati. Objave su sortirane po vremenu i datumu objave, od najnovije do najstarije. Na sličan način, postavlja se i upit za dobavljanje korisnika koje ulogovani korisnik prati. **Lista** objava i **lista** korisnika se pomoću Haskellove funkcije **zip** spaja u listu koja sadrži ***Tuple*** elemente. Pozivom funkcije **fst** nad elementom liste dobijamo objavu, dok pozivom funkcije **snd** nad elementom liste dobijamo korisnika koji je objavio tu objavu.

4. RAZLIČITI SCENARIJI KORIŠĆENJA INSTAGRAM KLON APLIKACIJE

4.1. Registracija i login korisnika

Na slici 5 prikazana je forma za unos podataka o korisniku. Klikom na dugme „Upload“ korisniku se otvara pretraživač fajlova, gde korisnik može da izabere i postavi svoju profilnu sliku. Klikom na dugme „Create User“ se vrši validacija forme i ako je sve u redu kreira se novi korisnik u bazi podataka. Šifra korisnika se hešuje tako da je nemoguće pročitati je u bazi podataka.

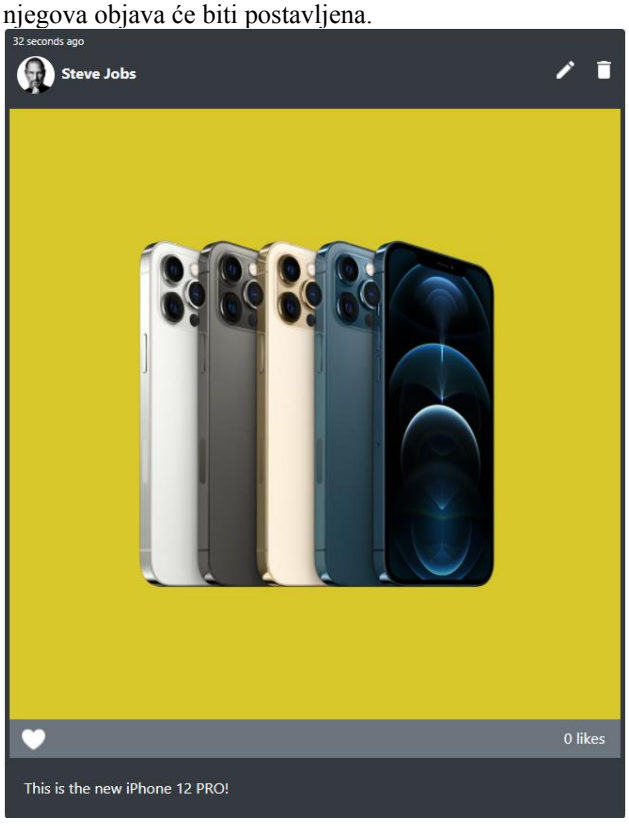


Slika 5. *Registracija korisnika*

Nakon registracije, korisnik je u mogućnosti da se uloguje u svoj nalog. Potrebno je da unese svoju e-mail adresu i šifru koju je uneo prilikom registracije. Klikom na dugme „Login“ korisnik će biti ulogovan.

4.2. Postavljanje objave

Korisnik je u mogućnosti da izabere sliku koju će se postaviti uz pomoć pretraživača fajlova, kao i da napiše željeni opis objave. Klikom na dugme „Create Post“ njegova objava će biti postavljena.

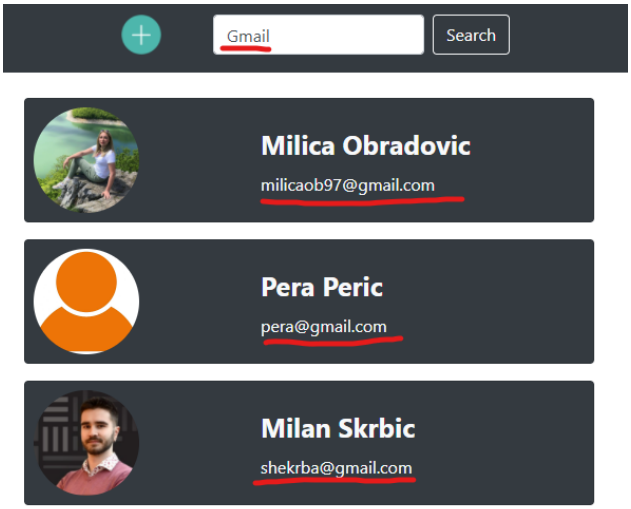


Slika 6. *Objava*

Na slici 6 prikazana je uspešno kreirana objava.

4.3. Prikaz rezultata pretrage korisnika

Moguće je uneti željeni tekst u polje za pretragu korisnika koje se nalazi u navigacionom baru. Nakon klika na „Search“ dugme prikazaće se svi korisnici koji imaju uneti tekst u imenu, prezimenu ili e-mail adresi. Prilikom pretrage korisnika nije bitno da li je uneti tekst za pretragu napisan velikim ili malim slovima (*engl. case-insensitive*).



Slika 7. *Rezultat pretrage*

Na slici 7 prikazan je rezultat pretrage korisnika na osnovu termina „Gmail“.

5. ZAKLJUČAK

Funkcionalno programiranje jeste noviji pristup rešavanju softverskih problema ali je zasnovano na teoriji koja već dugo postoji i sada napokon dobija veliku primenu u softverskom području. Sve veći broj stručnjaka se opredeljuje baš za ovaj pristup programiranju zato što vide veliki potencijal u njemu.

Međutim, funkcionalno programiranje je i dalje u veoma ranoj fazi razvoja i postoji dosta prostora za napredak i razvijanje raznih vrsta softverskih rešenja koje će pospešiti funkcionalno programiranje. Konkretno u IHP radnom okviru nije moguće korišćenje neke druge baze osim integrisane PostgreSQL baze podataka koju IHP pruža, kao ni korišćenje modernijih radnih okvira za klijentski deo aplikacije, kao što su ReactJS, Angular i VueJS.

Iako je ovaj pristup još mlad i stavljen na test, koji će u jednom trenutku dokazati da li ima poente pratiti ga ili ne, veliki broj vodećih softverskih kompanija, kao što su Facebook i Github, počeli su da ulažu u funkcionalno programiranje i određeni deo svojih softverskih rešenja su implementirali baš u Haskellu.

Funkcionalno programiranje i Haskell jesu zasnovani na velikoj apstrakciji koju pruža teorija kategorija. Baš zbog toga je teško programerima koji su navikli na imperativni pristup da pređu na funkcionalno programiranje, ali apstrakcija koju funkcionalno programiranje zahteva je trenutno neophodna da se prevaziđe veliki broj problema sa kojima su se programeri do sad susretali.

6. LITERATURA

[1] Bartosz Milewski, “ Category Theory for Programmers ” https://github.com/hmemcpy/milewski-ctfp-pdf#category-theory-for-programmers

[2] Philipp Hagenlocher, “ Haskell for Imperative Programmers ” https://www.youtube.com/watch?v=Vgu82wiiZ90&list=PLe7Ei6viL6jGp1Rfu0dil1JH1SHk9bgDV

[3] Shaistha Fathima, “ Functional Programming VS Object Oriented Programming (OOP) Which is better….? ” https://medium.com/@shaistha24/functional-programming-vs-object-oriented-programming-oop-which-is-better-82172e53a526

Kratka biografija:

Milan Škrbić rođen je u Novom Sadu 26.02.1996. god. Osnovne studije je završio na Fakultetu tehničkih nauka na smeru Računarstvo i Automatika 2019. god. Master studije je upisao iste godine na Fakultetu tehničkih nauka na smeru Računarstvo i Automatika, modul Elektronsko Poslovanje.

kontakt: shekrba@gmail.com

Зборник радова Факултета техничких наука, Нови Сад

UDK: 004.42:004.738.12
DOI: https://doi.org/10.24867/15BE37Dragisic

УПОТРЕБА TLS ПРОТОКОЛА ЗА БЕЗБЕДНУ КОМУНИКАЦИЈУ У SPRING РАДНОМ ОКВИРУ

USING TLS PROTOCOL FOR SECURE COMMUNICATION IN THE SPRING FRAMEWORK

Јелена Драгишић, *Факултет техничких наука, Нови Сад*

Област – ПРИМЕЊЕНЕ РАЧУНАРСКЕ НАУКЕ И ИНФОРМАТИКА

Кратак садржај – *Обрада особина TLS протокола, начина успостављања и прекида TLS везе. Акценат је стављен на HTTPS протокол, где су описане особине протокола, а начин функционисања је приказан кроз демонстрацију комуникације клијента и сервера која се одвија преко TLS везе.*

Кључне речи: *TLS, SSL, HTTPS, протокол*

Abstract – *Process the features of the TLS. Emphasis will be placed on the HTTPS protocol, where the properties of the protocol will be described, and the way how it works will be shown through the demonstration where the communication between client and server will be happened above TLS.*

Keywords: *TLS, SSL, HTTPS, protocol*

1. УВОД

Задатак овог рада јесте обрада *Transport Layer Security (TLS)* протокола. *TLS* протокол је развијен од стране међународне организације за стандарде, под називом *Internet Engineering Task Force (IETF)*, а прва верзија протокола објављена је 1999. године. Најновија верзија јесте *TLS* 1.3 и објављена је 2018. године. *TLS* протокол је сигурносни мрежни протокол који омогућава сигурну комуникацију на мрежи, сигурност података који се размењују и очување њиховог интегритета. Подржава и аутентификацију обе стране које учествују у комуникацији. Основна употреба овог протокола јесте шифровање комуникације између веб апликација и сервера.

У другом поглављу дат је кратак увод у криптографске протоколе, тачније описани су одређени безбедносни појмови који се користе у овом раду. Треће поглавље рада се односи на презентацију *TLS* протокола, начина успостављања и прекида сигурне комуникационе везе, а такође су описане предности и мане овог протокола. Комбинација *TLS*-а са *HTTP* протоколом презентована је у четвртом поглављу, које садржи и демонстрацију примене *TLS* протокола. Пето поглавље садржи смернице даљег развоја и представља закључак овог рада.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био доцент др Жељко Вуковић.

2. УВОД У КРИПТОГРАФСКЕ ПРОТОКОЛЕ

Криптографски протоколи представљају дефинисан скуп правила за решавање одређених ситуација као и спречавање потенцијалних проблема при самој комуникацији. *TLS* је управо криптографски протокол. С обзиром на то да ови протоколи почивају на коришћењу одређених криптографских техника, у овом поглављу је направљен увод у криптографске протоколе и објашњена су значења неких основних појмова који су коришћени кроз рад.

Пре свега, криптографија заправо представља научну дисциплину која обезбеђује очување тајности порука. Појам енкрипције се односи на шифровање података, односно превођење тих података из људски, разумљивог облика у низ насумичних карактера који не носе семантику. Декрипција је обрнут поступак, где се шифрован текст преводи у оригиналан. Сертификат представља на неки начин личну карту учесника комуникације. Обезбеђује могућност аутентификације, односно потврду идентитета самог учесника. Иза сваког валидног сертификата треба да стоји одређено сертификационо тело, које је тај сертификат заправо издало и потписало [1].

3. TLS ПРОТОКОЛ

TLS протокол је у *Open System Interconnection (OSI)* моделу смештен између петог и шестог слоја. Ради на слојевима изнад траспортних протокола попут *TCP* проткола, а испод апликацијских попут *HTTP* протокола [2]. Положај *TLS* протокола приказан је на слици 1.

Слика 1. *Положај TLS протокола*

TLS протокол је криптографски протокол који успоставља сигурну, комуникациону сесију између

пријемне и предајне стране. Основни задатак јесте очување приватности током комуникације и размене података као и заштита интегритета података који се траснпортују.

TLS је двослојни проткол, тачније састоји се из два подпроткола и то протокола руковања и протокола записа [3] . Током механизма руковања пријемна и предајна страна се договарају о начину шифровања комуникације, врши се аутентификација страна као и размена кључева. Затим, на договорен начин се криптују и на тај начин обезбеђују подаци који се преносе кроз *TLS* тунел, претходно успостављен између комуникационих страна.

Протокол руковања обезбеђује сигурност везе. Та веза је приватна а шифровање података који се размењују постижемо коришћењем симетричних криптографских алгоритама. Кључеви који се користе за енкрипцију генеришу се посебно за сваку конекцију у оквиру протокола руковања. Пре саме енкрпције подаци се компримују коришћењем договореног алгоритма. *TLS* веза је такође поуздана јер се при примању поруке проверава њен интегритет коришћем хеш вредности о чему ће више речи бити у наставку. Дакле, протокол руковања омогућује генерисање сигурне сесије између клијента и сервера, а протокол записа сигурну размену података. [4]

3.1. *TLS handshake* протокол

Успостава *TLS* комуникације између пријемне и предајне стране започиње процедуром руковања. Та процедура се назива *TLS handshake* протокол, а омогућава договарање клијента и сервера око пакета шифровања који ће бити коришћен у наставку комуникације. Пакет шифровања обухвата алгоритам за шифровање као и криптографске кључеве. Омогућена је и аутентификацијуа комуникационих страна, тачније потврду њиховог идентитета. Најчешће је то случај са серверском страном.

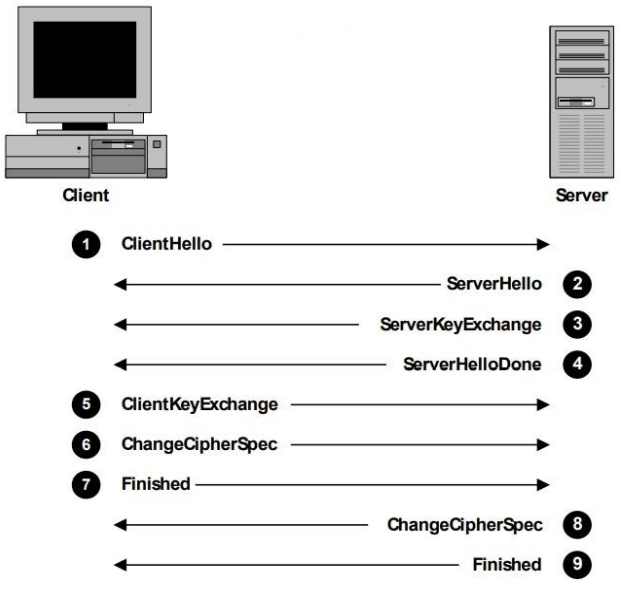
Клијент иницира комуникацију слањем *ClientHello* поруке серверу са којим жели да комуницира. У оквиру ове поруке специфира до које верзије има подршку за *TLS* проткол. Треба узети у обзир то да клијент подржава и раније верзије протокола закључно са оном која је специфирана. Порука садржи *RandomNumber* поље које садржи произвољну вредност дужине 32 бајта која се користи као основа за даља различита криптографска израчунавања. *SessionID* поље поруке је у иницијалној процедури руковања празно, али се користи у надоградњама овог механизма. Списак криптографских алгоритама као и дужине кључева које клијентска страна може да подржи документоване су у пољу *CipherSuites*. *CompressionMethods* поље садржи списак различитих компресионих метода које је са стране клијента могуће користити.

Након што прими ClientHello поруку, сервер одговара слањем ServerHello поруке. У тој поруци поставља верзију TLS протокола по ком ће се даља комуникација заснивати. Верзија протокола за коју се сервер одлучи мора бити у складу са верзијом коју је специфирао клијент у ClientHello поруци. Затим се постављају RandomNumber и SessionID поља. CipherSuite поље садржи неки од криптографских алгоритама са списка који је специфиран у CipherSuites пољу клијентске поруке. Сервер је та страна која доноси коначну одлуку о начинима енкрипције података током даље комуникације. На тај начин се комуникационе стране усклађују око пакета шифровања који ће користити. ServerKeyExchange порука садржи јавни кључ сервера који ће се користити за даљу енкрипцију, а формат кључа зависи од изабраног пакетог шифровања. Након ње, шаље се и ServerHelloDone порука која наговештава клијенту да може прећи у наредну фазу комуникације.

Након што добије јавни кључ од сервера, клијент је у могућности да пошаље ClientKeyExchange поруку. У овој поруци шаље податке које енкриптује јавним кључем сервера, а исте може само прочитати сервер ком је ова порука намењена.Међу подацима у овој поруци је и кључ сесије којим ће се даље шифровати подаци у овој комуникационој сесији.

Након слања *ClientKeyExchange* поруке, клијент серверу шаље и *ChangeCipherSpec* поруку којом указује на то да је успостављен или промењен пакет шифровања и да је даља комуникација обезбеђена на договорен начин. Клијент затим шаље и *Finished* поруку која супротној страни омогућава проверу испуњености договорених, сигурносних параметара у иницијалној фази комуникације.

Сервер одговара клијенту слањем *ChangeCipherSpec* и *Finished* поруке. Процес руковања се на овај начин успешно завршава, а даља комуникација је енкриптована. Поруке које се размењују током овог процеса, као и редослед истих, приказан је на слици 2.



Слика 2. *TLS handshake [5]*

Претходни модел успостављања *TLS* везе често је неопходно прошити. Тада је потребно обезбедити аутентификацију учесника комуникације, односно потврду њиховог идентитет. У већини случајева то је неопходно учинити за серверску страну. Потврду идентитета клијента могуће је постићи и провером самих података које клијент шаље током комуникације са сервером.

3.2. *TLS* протокол записа

TLS је двослојни проткол, тачније састоји се из два подпроткола и то протокола руковања и протокола записа [2].

TLS протокол записа омогућава сигурну размену података кроз успостављен *TLS* комуникациони канал. Сигурна размена почива на енкрипцији података употребом кључа и криптографског алгоритма. Користи се договорени пакет шифровања, који представља производ претходно описане процедуре руковања. Шифровани подаци се даље предају протоколима нижег нивоа на транспорт, конкретно протоколима транспортног слоја. Између осталог, протокол записа омогућава и проверу интегритета порука и на тај начин спречава нежељену измену података у транзиту.

3.3. Затварање TLS сесије

Када дође до тренутка затварања сесије, неопходно је исту експлицитно затворити. То се постиже слањем *ClosureAlert* поруке супротној комуникационој страни, док друга страна након пријема врши ретрансимисију поруке. На тај начин потврђено је затварање комуникационе сесије [5].

3.4. Предности и мане

Поред основних функција разматрано протокола, једна од главних предности јесте независност протокола која се огледа у употреби протокола вишег нивоа над *TLS* -ом без додатних, неопходних прилагођавања.

TLS протокол омогућава корисницима да безбедно приступе удаљеним ресурсима. Како би истима корисник приступио, неопходно је да се аутентификује. За тај поступак може се искористити TLS конекција. Приступ подацима је могуће ограничити.

Анализом *TLS* проткола долази се до закључка да је временски најзахтевнија фаза процес руковања између клијента и сервера [6]. Неопходна је размена одређеног броја порука у циљу постизања договора око пакета шифровања и успостављања сигурне, комуникационе сесије. Свако ново успостављање TLS везе између пријемне и предајне стране захтева понављање поступка руковања. То свакако представљаману протокола, али временом су креирани различити начини превазилажења ових и сличних недостатака.

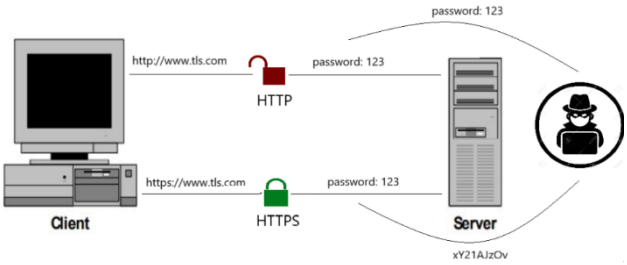
Такође, кеширање сесија на серверу представља потенцијални меморијски проблем за који се опет користе различите технике превазилажења

4. HTTPS ПРОТОКОЛ

HyperText Transfer Protokol (HTTP) је мрежни протокол апликативног слоја *OSI* референтног модела. Као идеја настаје 1989 године у Церну, али је прва верзија овог протокола *HTTP* V0.9 документована 1991. године. Архитектура протокола је клијент-сервер, односно сама комуникација се одвија између клијента и сервера, и то следећим низом трансакција. Клијент шаље поруку захтева, а сервер затим враћа поруку одговора. Порука захтева и порука одговора представљају два типа *HTTP* порука које се шаљу у отвореном тексту на мрежи. Свако ко прати комуникацију између две стране овакав вид отворених порука може да разуме. Ово представља озбиљан безбедносни проблем када се шаљу осетљиви подаци. Сам протокол се користи за обраду, приказивање и испоруку веб страница.

Како би се такви безбедносни ризици смањили, долази до развоја *HTTPS* протокола који представља шифровану, односно сигурну верзију *HTTP* протокола. Употребом *HTTPS* протокола обезбеђује се аутентификација крајњих, комуникационих тачака и поверљивост саме комуникације [7]. Овај протокол заправо представља употребу *HTTP* протокола над *TLS* -ом, где се *TLS* протокол користи за енкрипцију података у оквиру *HTTP* захтева и одговора. На овај начин малициозна трећа страна неће бити у могућности да види отворени текст већ само низ насумичних карактера.

Странице на вебу које користе *HTTP* протокол у оквиру свог *URL*-а садрже идентификатор проткола означен са “HTTP://“, док странице које користе *HTTPS* протокол садрже идентификатор “HTTPS://“. Поређење ова два протокола приказано је на слици 3.



Слика 3. *Приказ HTTP и HTTPS протокола*

Када се започиње ова врста комуникације, *HTTP* клијент се понаша као тлс клијент. Неопходно је иницирање комуникационе сесије са серверском страном претходно описаним протоколом руковања. Након што се успостави сигурни тлс комуникациони канал, могуће је слање првог *HTTP* захтева [8]. Сви подаци који се преносе путем ових захтева су енкриптовани, док се њихов интегритет проверава. Битан корак за успостављање сигурне *HTTPS* везе јесте провера идентитета сервера. Због тога најважнији део подешавања окружења јесте коришћење сертификата.

4.1. Spring радни оквир

Spring је open source развојно окружење које се користи за развој апликација коришћењем *Java* програмског језика. Садржи екстензију *Spring Boot* која са собом носи бројне погодности у виду руковања конфигурацијом, како би се скренуо фокус на развој саме апликације.*Spring Boot framework* је коришћен за потребе демонстрације истраживања.

4.2. Демонстрација истраживања

За потребе задатка направљена је *Spring Boot* апликација која неће имати никакве

функционалности, већ ради једноставности само један endpoint.

Како бисмо омогућили *HTTPS* у нашој апликацији било је неопходно креирање сертификата.

Сертификат је генерисан путем терминала за шта нам је био потребан алат *keytool*, који можемо преузети са интернета [9] или не морамо уколико имамо инсталиран JDK (*Java Development Kit*), јер се поменути алат налази у оквиру JDK-а. Сертификате можемо генерисати користећи *Let's encrypt* [10]. То је непрофитан *certificate authority (CA)* који нам пружа X.509 сертификате [11] за *Transport Layer Security (TLS)*, без накнаде. То је највећи светски *CA* чији је главни циљ да вебсајтови постану сигурни и да користе *HTTPS*.

Након што се обезбеди сертификат извршена је конфигурација *HTTPS*-а у *Spring Boot*-у. Поред тога било је потребно да извршимо редирекцију на *HTTPS*, уз помоћ *Spring Security*-а. Када користимо *Spring Security* тада можемо да конфигуришемо сервер тако да се аутоматски блокира било који *request* који долази са небезбедног *HTTP* канала и редиректује на *HTTPS*.

5. ЗАКЉУЧАК

Циљ рада био је да се осврнемо на *TLS* протокол, на његове особине а пре свега на начин успостављања, као и прекида конекције. Размотрене су предности, али свакако и мане протокола. Акценат је стављен на *HTTPS* протокол, чије су особине такође описане, а начин функционисања је приказан кроз демонстрацију истраживања.

Експанзијом комуникације на мрежи, сама сигурност података који се размењују добија велики значај. Интернет мрежа постаје водећа инфраструктура када је у питању комуникација, трговина и генерални приступ информацијама. Подаци који се преносе се у свом изворном облику су под великим безбедносним ризиком. Заштиту таквих података који се транспортују путем мреже омогућава управо примена *TLS* протокола. Данас, готово сви веб претраживачи и сервери имају уграђену подршку за овај протокол у циљу обезбеђивања сигурног преноса података између комуникационих страна.

6. ЛИТЕРАТУРА

[1] Uvod u kriptografiju – Bezbednost u sistemima elektronskog poslovanja, Prof. dr Goran Sladić

[2] A survey on MITM and its countermeasures in the TLS handshake protocol, Seung-Woo Han, Hyunsoo Kwon, Changhee Hahn, Dongyoug Koo, Junbeom Hur , Department of Computer Science and Engineering, Korea University, Seoul, Republic of Korea

[3] An Analysis of TLS Handshake Proxying, Douglas Stebila, Nick Sulivan

[4] *TLS protokol CCERT-PUBDOC-2009-03-257*, Hrvatska akademska i istraživačka mreža

[5] SSL & TLS Essentials Securing the Web, Stephen A. Thomas

[6] Design of an enhancement for SSL/TLS protocols, Ashraf Elgohary, Tarek S. Sobh, M. Zaki

[7] The Cost of the "S" in HTTPS, David Naylor, Alessandro Finamore, Ilias Leontiadis, Yan Grunenberger, Marco Mellia, Maurizio Munafò, Konstantina Papagiannaki, Peter Steenkiste

[8] *RFC 2818– HTTP over TLS*, E. Rescorla

[9] Spring Initializr, *(https://start.spring.io/)*

[10] Let's encrypt, *(https://letsencrypt.org//)*

[11]X.509 сертификат, (*https://en.wikipedia.org/wiki/X.509*)

Кратка биографија:

Јелена Драгишић рођена је у Новом Саду 1998. године. Мастер рад на Факултету техничких наука из области Електротехнике и рачунарства – Примењене рачунарске науке и информатика одбранила је 2021. године.

контакт: jelenaa.dragisic@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.738.1:766
DOI: https://doi.org/10.24867/15BE38Tarjan

SISTEM ZA DISTRIBUIRANU DETEKCIJU PLAGIJARIZMA

DISTRIBUTED PLAGIARISM DETECTION SYSTEM

Tamaš Tarjan, *Fakultet tehničkih nauka, Novi Sad*

Oblast – SOFTVERSKO INŽENJERSTVO I INFORMACIONE TEHNOLOGIJE

Kratak sadržaj – *U radu je predstavljen sistem za distribuiranu detekciju plagijarizma. Sistem nudi centralizovani pristup distribuiranim čvorovima koji su prilagođeni protokolu za distribuiranu proveru plagijarizma. U ovom radu je dat opis navedenog protokola. Distribuirani čvorovi imaju mogućnost da se priključe u sistem pomoću registra servisa. Prilikom registracije čvora u sistem, izvršava se osnovna provera poštovanja protokola simulirajući očekivani tok komunikacije između datog čvora i sistema. Svaki čvor ima svoje skladište za digitalne dokumente. Dokumenti nisu deljeni između članova sistema.*

Ključne reči: *Distribuirani sistem, detekcija plagijarizma, obrada dokumenata*

Abstract – *This paper describes a system for distributed plagiarism detection. The system offers centralized access to distributed nodes that adhere to a protocol for distributed plagiarism detection that is described in this paper. Distributed nodes can join the system through service discovery. Adherence to the mentioned protocol is verified during node registration by simulating the expected request / response communication flow. Each node has its own digital document storage implementation. Documents are not shared between the system nodes.*

Keywords: *Distributed system, plagiarism detection, document processing*

1. UVOD

Plagijarizam predstavlja potpunu ili delimičnu upotrebu tuđih dela i misli, i predstavljanje istih kao sopstvene ideje.

U današnje vreme sve je veći broj domena u kojima postoji potreba za skladištenjem neke vrste digitalne intelektualne svojine. Samim tim broj ustanova koji imaju svoje privatne sisteme za skladištenje navedenog digitalnog sadržaja je velik. Data skladišta digitalnih intelektualnih svojina mogu biti javna ili privatna.

Jedan od glavnih problema prilikom utvrđivanja plagijata je to što je broj izvora iz kojih može da potiče deo teksta koji je plagiran previše velik.

Sistemi za proveru plagijarizma imaju pristup samo relativno malom podskupu relevantnog sadržaja.

Određeni broj ustanova pored skladištenja digitalnih dokumenata takođe imaju mogućnost provere plagijarizma u okviru njihovog skupa dokumenata.

Distribuirani sistem je sistem čije komponente se nalaze na različitim umreženim računarima koji komuniciraju i orkestriraju svoje akcije pomoću slanja poruka jedni drugima [1].

U ovom radu je predstavljen sistem koji pruža mogućnost postojećim skladištima dokumenata da se priključe u distribuirani anonimizovan sistem za detekciju plagijarizma bez deljenja svojih digitalnih dokumenata.

2. TEORIJSKE OSNOVE

Način komunikacije između čvorova u distribuiranom sistemu zavisi od topologije datog sistema. U ovom sistemu postoji jedan centralni čvor koji komunicira sa ostalim učesnicima i omogućuje korisniku da vrši proveru plagijarizma na proizvoljnim čvorovima sistema. U narednom delu su predstavljeni ključni pojmovi za opis ovog sistema.

2.1. Dokument

U ovom radu pojam dokumenta se definiše kao digitalna jedinica obrade koja predstavlja intelektualnu svojinu.

2.2. Metapodaci dokumenta

Metapodaci dokumenta predstavljaju podatke o dokumentu. Primeri metapodataka su naziv dokumenta, autori, tip dokumenta, veličina dokumenta, lokacije različitih reprezentacija dokumenta, itd.

2.3. Reprezentacija dokumenta

Format originalne reprezentacije dokumenta koja se skladišti nakon uspešnog otpremanja na sistem, ne mora da bude kompatibilna sa određenim algoritmom za detekciju plagijarizma, bez obzira da li konceptualni tip dokumenta odgovora datom algoritmu (na primer tekstualni dokument u PDF formatu).

Da bi detekcija plagijarizma za dati dokument bila moguća, treba kreirati određenu reprezentaciju sa kojom algoritam za detekciju plagijarizma može da rukuje.

2.4. Predprocesiranje dokumenata

Predprocesiranje dokumenta predstavlja proces u kojem se na osnovu ulaznog dokumenta dobija određeni izlaz. Izlaz predprocesiranja može biti određena reprezentacija dokumenta ili metapodatak koji je dobijen na osnovu ulaznog dokumenta.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Ivanović, red. prof.

2.5. Detekcija plagijarizma

Detekcija plagijarizma predstavlja proces u kojem se kandidat dokument upoređuje sa svakim dokumentom iz referentnog skupa dokumenata. Referentni dokumenti su dokumenti koji pripadaju referentnom skupu. U slučaju da postoji značajno preklapanje između kandidata i referentnog dokumenta, kandidat se markira kao potencijalni plagijat.

2.6. Referentni skup

U idealnom slučaju referentni skup je skup svih relevantnih dokumenata sa kojima treba uporediti kandidat dokument da bi utvrdili da li je dati dokument plagijat. Ovaj skup možemo nazvati apsolutnim referentnim skupom. U praktičnom smislu nemamo mogućnost da pristupimo svim dokumentima iz ovog skupa, to znači da je suštinski nemoguće dokazati da određeni dokument nije plagijat. Iz tog razloga fokus je na dokazivanju da dati dokument jeste plagijat, zato što je to potencijalno izvodljiv zadatak. Skup koji sadrži dokumente sa kojima upoređujemo kandidat dokument možemo nazvati realnim referentnim skupom (ili samo referentnim skupom).

Ako pretpostavimo da imamo algoritam koji sa sigurnošću upoređuje dva data dokumenta i daje procenu da li ja kandidat dokument plagijat, tada verovatnoća da se detektuje plagijarizam kod datog dokumenta zavisi od broja dokumenata u referentnom skupu i od broja dokumenata koji su bili plagirani u datom kandidat dokumentu. Ovu verovatnoću možemo maksimizovati maksimizacijom broj dokumenata u referentnom skupu.

3. SPECIFIKACIJA SISTEMA

U ovom poglavlju je opisana specifikacija centralizovanog sistema koja komunicira sa krajnjim čvorovima koji vrše proveru plagijarizma u okviru svog referentnog skupa dokumenata. Čvorovi za detekciju plagijarizma moraju da se prilagode protokolu koji je opisan u nastavku.

3.1. Protokol za distribuiranu detekciju plagijarizma

Protokol za distribuiranu detekciju plagijarizma posmatra krajnje čvorove kao REST resurse. Komunikacija sa datim čvorovima se vrši isključivo preko HTTP prokola upotrebom JSON poruka. Krajnje tačke koje čvorovi moraju da podrže su navedene na listinzima 3.1.1. - 3.1.3.

```
Request: GET <BASE_URL>/api/v1/algorithms
Response:
[
    {
        "id": "1",
        "name": "<algorithm-name>",
        "description": "<algorithm-description>",
        "supportedMimeTypes": [
            "text/plain"
        ]
    }
]
```

Listing 3.1.1. *Zahtev za dobavljanje podržanih algoritama za detekciju plagijarizma*

```
Request: POST <BASE_URL>/api/v1/tasks HTTP/1.1
Content-Type: multipart/form-data; boundary=----FormBoundary
Content-Length: <content-length>
----FormBoundary
Content-Disposition: form-data; name="file"; filename="file-path"
Content-Type: application/pdf
(data)
----FormBoundary
Content-Disposition: form-data; name="algorithmId"
(algorithm-id)
----FormBoundary
Response:
{
    "id": "<task-id>",
    "status": "<status>",
    "algorithmId": "<algorithm-id>",
    "candidateDocumentId": "<id assigned by the node>",
    "candidateDocumentMd5": "<MD5 string>"
}
```

Listing 3.1.2. *Zahtev za kreiranje zadatka za proveru plagijarizma*

```
Request: GET /api/v1/task/<taskId>
Response:
{
    "id": "<task-id>",
    "status": "<status>",
    "algorithmId": "<algorithm-id>",
    "candidateDocumentId": "<id assigned by node>",
    "candidateDocumentMd5": "<MD5 string>",
    "results": [
        {
            "referenceDocumentId": "<id assigned by node>",
            "referenceDocumentMd5": "<MD5 string>",
            "score": <similarity-score>
        }
    ]
}
```

Listing 3.1.3. *Zahtev za proveru statusa zadatka za proveru plagijarizma*

Rezultati koji se mogu dobiti prilikom provere statusa zadatka ne sadrže konkretne informacije o dokumentu osim identifikatora koji je dodeljen dokumentu od strane čvora i MD5 (eng. Message-digest algorithm) checksum vrednosti radi identifikacije dokumenta. Čvorovi mogu opciono da vrate listu sličnih delova dokumenata za svaki rezultat, međutim i u tom slučaju šalju samo MD5 checksum vrednost od datih delova. Ako se utvrdi da postoje slični dokumenti onda su navedene checksum vrednosti dovoljne da se plagirani dokumenti identifikuju.

Čvorovi moraju da podrže *simulate* parametar (eng. query parameter) za svaki zahtev koji se prosleđuje prilikom testiranja čvora. U slučaju da je ovaj parametar naveden u nekom zahtevu koji je specificiran od strane protokola čvor treba da vrati validan tip podataka za svako polje koje može da se pojavi u odgovoru na zahtev.

3.2. Servis za detekciju plagijarizma

Servis za detekciju plagijarizma predstavlja primer implementacije čvora koji je prilagođen protokolu za distribuiranu proveru plagijarizma. Pruža mogućnost za skladištenje tekstualnih dokumenata i njihovih metapodataka kao i osnovno predprocesiranje koje je potrebno za detekciju plagijata tekstualnih dokumenata. Čvor predstavlja serversku stranu klijent-server komunikacije koja opslužuje zahteve specificirane protokolom koji je opisan u poglavlju 3.1.

3.3. Registar servisa za detekciju plagijarizma

Registar servisa je veb aplikacija koja pruža mogućnost čvorovima da se registruju za učestvovanje u procesu distribuirane provere plagijarizma. Kada neki čvor pošalje zahtev za registraciju registar vrši simulirani tok komunikacije sa datim čvorom da bi utvrdio validnost datog čvora. U slučaju da čvor ne poštuje minimalne zahteve prilikom simuliranog toka komunikacije, registracija čvora se otkazuje.

Registar servisa pruža mogućnost centralizovanom servisu za detekciju plagijarizma da dobije lokacije registrovanih čvorova i da uspostavi komunikaciju sa datim čvorovima.

3.4. Servis za upravljanje distribuiranim zadacima provere plagijarizma

Zadatak ovog servisa je da ponudi objedinjeni interfejs odnosno API (eng. Application Programming Interface) za upravljanje distribuiranim zadacima provere plagijarizma. Ovaj servis predstavlja klijenta koji komunicira sa krajnjim čvorovima upotrebom protokola koji je opisan u poglavlju 3.1.

Ovaj servis obezbeđuje korisniku sledeće funkcionalnosti: pregled liste čvorova koji su u određenom trenutku dostupni za izvršavanje distribuirane detekcije plagijarizma, kao i liste algoritama koje navedeni čvorovi nude. Pored toga, servis nudi krajnje tačke za upravljanje distribuiranim zadacima. Ovo podrazumeva kreiranje zadataka koji su deljeni između proizvoljnog broja čvorova kao i proveru statusa i eventualnih rezultata datih zadataka. Zadaci kreirani od strane jednog korisnika u okviru ovog servisa su dostupni samo datom korisniku.

3.5. Korisnički interfejs

Korisnički interfejs nudi funkcionalnosti za registraciju i prijavu korisnika, pregled dostupnih čvorova za detekciju plagijarizma kao i kontrole za upravljanje zadacima detekcije plagijarizma. Korisnički interfejs zajedno sa servisom za upravljanje distribuiranim zadacima provere plagijarizma pruža mogućnost da se na jednostavan način izvrši otpremanje kandidat dokumenta na odabrane krajnje čvorove koji će vršiti proveru plagijarizma na svojim referentnim skupovima. Korisnik ima mogućnost pregleda rezultata u okviru korisničkog interfejsa.

4. IMPLEMENTACIJA SISTEMA

U ovom poglavlju je dat opis implementacije sistema za upravljanje distribuiranim zadacima detekcije plagijarizma kao i primer sistema koji predstavlja jednu implementaciju protokola koji je opisan u trećem poglavlju ovog rada.

Servis za upravljanje distribuiranim zadacima provere plagijarizma, kao i primer krajnjeg čvora koji vrši proveru plagijarizma su implementirani u obliku veb aplikacija pomoću Java programskog jezika upotrebom Spring Boot radnog okvira.

4.1. Servis za upravljanje distribuiranim zadacima provere plagijarizma

Da bi izbegli potrebu da *frontend* aplikacija odnosno korisnički interfejs komunicira direktno sa pojedinačnim čvorovima koji učestvuju u distribuiranom sistemu, implementiran je servis koji nudi jednostavan API za lako upravljanje distribuiranim procesom za detekciju plagijarizma uz oslonac na *Backend for frontends* šablon. *Backend for frontends* šablon se koristi u kontekstu modernih mikroservisa za objedinjavanje različitih API-a koje odvojeni mikroservisi nude [9]. Sličan pristup je upotrebljen prilikom implementacije ovog servisa.

Entiteti u okviru ovog servisa su skladišteni u MongoDB bazi podataka. Entitet koji predstavlja jedan distribuirani zadatak detekcije plagijarizma u okviru ovog servisa se sastoji od niza podzadataka koji su asocirani sa pojedinačnim čvorovima koji vrše detekciju plagijarizma. Podzadaci su asocirani sa kandidat dokumentom preko njegove MD5 checksum vrednosti.

Kada korisnik kreira distribuirani zadatak, ovaj servis šalje po jedan zahtev čvorovima koji su specificirani u korisnikovom zahtevu. Ovim zahtevom krajnji čvorovi dobijaju sve potrebne informacije da bi izvršili traženi zadatak detekcije plagijarizma.

Kada korisnik pošalje zahtev za proveru statusa i rezultata, ovaj servis po potrebi šalje zahtev krajnjim čvorovima i prikuplja status i rezultate datih podzadataka nakon čega vraća objedinjeni odgovor korisniku.

4.1.1. Registar servisa za detekciju plagijarizma

Registar servisa je veb aplikacija implementirana uz oslonac na S*erver-side Service Discovery* šablon. Naziv, opis i lokacija čvorova za detekciju plagijarizma koji se registruju u registar servisa se skladišti u MongoDB bazi podataka. Servis za upravljanje distribuiranim zadacima za detekciju plagijarizma ima mogućnost da dobije navedene podatke o čvorovima koji su dostupni u mreži.

4.1.2. Korisnički interfejs

Korisnički interfejs je implementiran u Java 1.8 programskom jeziku upotrebom Vaadin 21.0.0.beta2 radnog okvira za izradu *frontend* aplikacija. Rezultati sadrže eksterne identifikatore referentnih dokumenata i kandidat dokumenta, kao i njihovu MD5 checksum vrednost. Navedeni identifikatori su dodeljeni od strane čvorova koji su vršili proveru.

4.2. Primer čvora za detekciju plagijarizma

U ovom delu je opisana implementacija primera krajnjeg čvora koji ima mogućnost da se priključi distribuiranom sistemu za proveru plagijarizma.

Prilikom pokretanja servisa automatski se izvršava pokušaj prijave na registar servisa za detekciju plagijarizma. Lokacija registra se definiše pomoću konfiguracionog parametra. Primer zahteva za prijavu na registar servisa je prikazan na listingu 4.2.1.

```
Request: POST <BASE_URL>/api/v1/services
Content-Type: application/json
{
    "port": 7771,
    "description": "Primer cvora za detekciju plagijarizma v1.0.0"
}
```

Listing 4.2.1. *Primer zahteva za prijavu na registar servisa*

4.2.1. Skladištenje dokumenata i metapodataka

Ovaj servis podržava obradu i skladištenje tekstualnih dokumenata predstavljenih PDF datotekom ili datotekom koja sadrži niz kodiranih karaktera pomoću UTF-8 (eng. Unicode Transformation Format) [2] standarda za kodiranje karaktera.

Prilikom otpremanja, servis skladišti originalnu reprezentaciju dokumenta kojoj se dodeljuje novi naziv u obliku UUID (eng. Universally Unique Identifier) vrednosti, nakon čega započinje proces obrade.

4.2.2. Obrada dokumenata

Obrada jednog dokumenta u okviru ovog servisa podrazumeva utvrđivanje formata datoteke. Format datoteke se predstavlja *MIME type* identifikatorom formata datoteki. Validne vrednosti *MIME type* identifikatora su definisane od strane *IANA* (eng. Internet Assigned Numbers Authority) organizacije [3]. Detektovanje *MIME type* identfikatora se vrši pomoću *Apache Tika* Java [4] biblioteke. U

slučaju da je originalna reprezentacija dokumenta predstavljena PDF datotekom, servis vrši ekstrakciju tekstualnog sadržaja pomoću *Apache PDFBox* [5] Java biblioteke. Ekstraktovani tekst se skladišti u zasebnoj datoteci čiji sadržaj je kodiran pomoću *UTF-8* kodiranja. Lokacija ekstraktovane reprezentacije se čuva u skupu metapodataka otpremljenog dokumenta. Metapodaci o dokumentu se čuvaju u MongoDB bazi podataka.

4.2.3. Implementacija protokola za distribuiranu detekciju plagijarizma

U skladu sa protokolom opisanim u trećem poglavlju ovog rada, implementirane su sve krajnje tačke koje su potrebne ovom servisu kako bi mogao da se priključi u distribuiranu mrežu za detekciju plagijarizma. Priključenje u distribuiranu mrežu uključuje i podršku da se simulira tok komunikacije sa ciljem da se utvrdi kompatibilnost prema datom protokolu. Simulirani odgovori na zahteve koji su poslati sa *simulate* parametrom su generisani uz pomoć *PODAM (eng. Pojo Data Mocker)* Java biblioteke [6] koja pruža mogućnost nasumičnog generisanja vrednosti za polja koja se mogu naći u odgovoru na određene zahteve koji su specificirani od strane protokola. Vrednosti u poljima odgovora će uvek biti ispravnog tipa. Za polja koja su tipa *JSON* niza se generiše po pet slučajnih elemenata.

4.2.4. Detekcija plagijarizma

Algoritam za proveru plagijarizma između tekstualnih datoteka je implementiran u Python 3.9.5 programskom jeziku pomoću *Scikit-learn* [7] biblioteke uz oslonac na *TfIdf* (eng. Term frequency-inverse document frequency) vektorizaciju i kosinusnu sličnost (eng. Cosine Similarity) između dobijenih vektora koji predstavljaju parove upoređenih dokumenata.

U ovom radu je fokus na distribuiranoj arhitekturi za detekciju plagijarizma i shodno tome se koristi pojednostavljena verzija algoritma koji je opisan u radu [8] gde su autori predstavili pristup za detekciju plagijarizma koji je baziran na upotrebi *TfIdf* vektorizacije i kosinusne sličnosti. Čvorovi koji implementiraju algoritme sa boljim performansama za veći broj tipova dokumenata se mogu na lak način priključiti u sistem.

5. ZAKLJUČAK

Broj digitalnih dokumenata kao i broj različitih skladišta za date dokumente svake godine raste zbog uvođenja digitalizacije u veliki broj domena u kojima se rukuje dokumentima. U ovom radu je predstavljen sistem koji za cilj ima objedinjavanje referentnih skupova dokumenata koji učestvuju u detekciji plagijarizma radi povećanja pokrivenosti apsolutnog referentnog skupa prilikom provere plagijarizma.

5.1. Dalji razvoj sistema

Trenutna implementacija sistema pruža mogućnost automatizovane distribuirane provere plagijarizma, međutim u slučaju da se pronađu potencijalni plagirani dokumenti, ne postoji proces pomoću kojeg bi moglo da se manuelno proveri da li su dati dokumenti zaista bili plagirani. Razlog ovome je zahtev da referentni dokumenti koji učestvuju u distribuiranoj detekciji plagijarizma ostanu privatni. Jedan način da se ovaj problem delimično reši je da protokol za distribuiranu proveru plagijarizma podrži deljenje podskupa dokumenata iz referentnog skupa. Vlasnici pojedinačnih čvorova bi trebali da odluče koji dokumenti su javni a koji privatni. Ovaj pristup i daje ima nedostatak kada neki čvor detektuje sličnost u privatnom referentnom dokumentu. Jedan od načina da se ovaj problem reši je uvođenje poslovnog procesa u kojem učesnici na određeni način ručno verifikuju da li je dati dokument plagiran.

6. LITERATURA

[1] Van Steen, M. and Tanenbaum, A., 2002. Distributed systems principles and paradigms. Network, 2, p.28.

[2] Allen, J.D., Anderson, D., Becker, J., Cook, R., Davis, M., Edberg, P., Everson, M., Freytag, A., Iancu, L., Ishida, R. and Jenkins, J.H., 2012. The unicode standard. Mountain view, CA.

[3] Iana.org. (2019). Media Types. [online] Dostupno na: https://www.iana.org/assignments/media-types/media-types.xhtml. [Pristupljeno 11. 8. 2021]

[4] tika.apache.org. (n.d.). Apache Tika – Apache Tika. [online] Dostupno na: https://tika.apache.org/. [Pristupljeno 11. 8. 2021]

[5] pdfbox.apache.org. (n.d.). Apache PDFBox | A Java PDF Library. [online] Dostupno na: https://pdfbox.apache.org/. [Pristupljeno 11. 8. 2021]

[6] mtedone.github.io. (n.d.). Podam - Welcome to Jemos PODAM (POjo DAta Mocker). [online] Dostupno na: https://mtedone.github.io/podam/. [Pristupljeno 11. 8. 2021]

[7] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, pp.2825-2830.

[8] Chavan, H., Taufik, M., Kadave, R. and Chandra, N., 2021. Plagiarism Detector Using Machine Learning. International Journal of Research in Engineering, Science and Management, 4(4), pp.152-154.

[9] Brown, K. and Woolf, B., 2016, October. Implementation patterns for microservices architectures. In Proceedings of the 23rd Conference on Pattern Languages of Programs (pp. 1-35).

Kratka biografija:

Tamaš Tarjan rođen je 6. decembra 1996. godine u Novom Sadu. Godine 2015. upisao je Fakultet tehničkih nauka u Novom Sadu smer Softversko inženjerstvo i informacione tehnologije. 2019. godine diplomirao je na osnovnim studijama na Fakultetu tehničkih nauka, iste godine upisuje master studije na smeru Softversko inženjerstvo i informacione tehnologije.

Kontakt: tarjan.nt@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.738.1:766
DOI: https://doi.org/10.24867/15BE39Zeljkovic

PROŠIRIVI SISTEM ZA PROVERU PLAGIJARIZMA BAZIRAN NA KOMPONENTAMA

EXTENSIBLE COMPONENT BASED PLAGIARISM DETECTION SYSTEM

Nikola Zeljković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – SOFTVERSKO INŽENJERSTVO I INFORMACIONE TEHNOLOGIJE

Kratak sadržaj – *U radu je opisan proširiv sistem za proveru plagijarizma na skupovima dokumenata različitih tipova. Sistem pruža mogućnost veoma lakog dodavanja komponenti za: proveru plagijarizma, pristup referentnom skupu dokumenata i predprocesiranje dokumenata. Prethodna podela komponenti daje mogućnost provere plagijarizma širokog spektra različitih tipova dokumenata.*

Ključne reči: *Detekcija plagijarizma, obrada dokumenata, proširiva arhitektura, plugin*

Abstract – *The goal of this paper is to present an extensible component based system for plagiarism detection. The system can be easily extended using components for: plagiarism detection, accessing reference sets and document processing. Using these components allow for performing plagiarism detection on a wide range of document types.*

Keywords: *Plagiarism detection, document processing, extensible arhitecture, plugin*

1. UVOD

Pojam plagijarizma se definiše kao upotreba i predstavljanje tuđih tekstova, umetničkih radova, izvornih kodova pisanim u različitim programskim jezicima, itd. kao ličnu intelektualnu svojinu. Ovde spada falsifikovanje tuđeg dela i rada čime se krši niz autorskih prava. Pored različitih ljudskih delatnosti veliku pažnju privlači plagijarizam u naučnoj i obrazovnoj oblasti [1].

Do velike rasprostranjenosti plagijarizma doveo je ubrzan razvoj tehnologije i mogućnosti koje ona pruža. U današnje vreme ljudi imaju mogućnost da na veoma jednostavan i brz način, upotrebom interneta dođu do velikog broja različitih izvora informacija. Postoji veliki broj softverskih rešenja za proveru plagijarizma koji daju mogućnost analize naučnih radova i drugih tekstualnih dokumenata.

U naučnom domenu postoji određeni broj tipova dokumenata za koje je poželjno vršiti proveru plagijarizma. U ovom radu je predstavljen sistem koji za cilj ima da pruži mogućnost provere plagijarizma za sve tipove dokumenata koji imaju značaj u određenim domenima gde pojam intelektualne svojine igra bitnu ulogu.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Ivanović, red. prof.

Posmatranjem domena u kojem se stvaraju tekstualni dokumenti i softverska rešenja predstavljena pomoću izvornih kodova, poželjno je vršiti proveru plagijarizma za date tipove dokumenata. U slučaju da se u ovom domenu pojavi potreba za podržavanjem trećeg tipa dokumenta, bilo bi poželjno da postoji mogućnost lakog proširenja postojećeg sistema za detekciju plagijarizma da podrži novi tip dokumenta. U ovakvom sistemu ne postoji potreba da se implementira ili konfiguriše potpuno novi sistem radi podržavanja novog tipa dokumenta, već je dovoljno samo dodati komponente koje će rukovati sa novim tipom dokumenta.

2. TEORIJSKE OSNOVE

Za implementaciju lako proširivog sistema potrebno je razmotriti sam proces koji se automatizuje upotrebom datog sistema. Da bi se izvršila automatizovana provera plagijarizma između jednog dokumenta (eng. Candidate document) i skupa referentnih dokumenata (eng. Reference documents) potrebno je izvršiti predprocesiranje datih dokumenata, nakon čega algoritam za proveru plagijarizma pristupa datim dokumentima radi utvrđivanja potencijalnog plagijarizma.

2.1. Dokument

U ovom radu pod pojmom dokumenta podrazumeva se pojam digitalnog dokumenta. Digitalni dokument predstavlja računarski obrađenu informaciju kojom se rukuje kao osnovnom jedinicom obrade.

U ovom radu se spominju sledeći tipovi dokumenata:

- Tekstulani dokumenti u PDF formatu
- Izvorni kodovi u programskom jeziku Python

2.2. Metapodaci dokumenata

Metapodaci dokumenata predstavljaju podatke o dokumentima. Osnovni metapodaci koji su bitni za ovaj sistem su naziv, tip i putanja do reprezentacije dokumenta koja će se koristiti prilikom izvršavanja provere plagijarizma. Dokument sam po sebi ne sadrži sve potrebne informacije za skladištenje, uspostavljanje veze sa drugim entitetima i proveru plagijarizma datog dokumenta. Ove informacije su rezultat predprocesiranja u čuvaju se u skupu metapodataka.

2.3. Predprocesiranje dokumenata

Predprocesiranje dokumenta je proces u kojem se izvršava niz operacija nad datim dokumentom. Svaka operacija proizvodi određenu reprezentaciju datog dokumenta i/ili metapodatak koji se skladišti u skupu metapodataka datog dokumenta. Određene operacije se mogu izvršiti nad dokumentom samo nakon što je predprocesiranje datog dokumenta završeno.

2.4. Skladištenje dokumenata i metapodataka

U okviru jednog sistema koji se delom bavi skladištenjem dokumenata, metapodaci dokumenata mogu se čuvati na različite načine. Originalna reprezentacija, kao i ostale reprezentacije datog dokumenta se mogu čuvati na fajl sistemu ili uz pomoć određenog eksternog rešenja za skladištenje dokumenata. Kompanije koje nude usluge u računarskom oblaku imaju servise za čuvanje masivnih količina podataka. Jedan od najpopularnijih primera za ovakvu vrstu servisa je Amazonov servis S3 [2]. Lokacije različitih reprezentacija određenog dokumenta se čuvaju u metapodacima datog dokumenta.
Skup dokumenata sa kojima se može vršiti provera plagijarizma može da postoji i van specijalizovanog sistema za skladištenje dokumenata, npr. dokumenti u određenom direktorijumu jednog računara.

2.5. Detekcija plagijata

Postoji veliki broj tipova dokumenata koji mogu biti kandidati za proveru plagijarizma. Pojedinačni pristupi za proveru plagijarizma su prilagođeni tipovima dokumenata za koje su implementirani. Postoji značajna razlika u pristupu prilikom utvrđivanja plagijarizma teksta, umetničkog dela ili izvornog koda određenog programskog jezika. Postoji razlika čak i ako posmatramo dva izvorna koda pisana različitim programskim jezicima. Za svaki pristup čija upotreba se razmatra, treba utvrditi koji tipovi dokumenata kao i koje reprezentacije dokumenta su potrebne za primenu datog pristupa.

2.6. Arhitektura bazirana na komponentama

Arhitektura bazirana na komponentama nam pruža mogućnost lakog dodavanja novih komponenti u sistem [3] radi podržavanja različitih skladišta dokumenata kao i različite algoritme za detekciju plagijarizma. Poželjno je omogućiti dodavanje novih komponenti bez privremenog gašenja sistema da bi vreme za koje sistem nije dostupan sveli na minimum.

2.6.1. Visual Studio Code

Visual Studio Code (VSCode u nastavku) je univerzalni editor izvornih kodova proizveden od strane Microsoft-a [4], i odličan primer softvera čija arhitektura je zasnovana na komponentama. VSCode pruža mogućnost instalacije ogromnog broja različitih ekstenzija za podršku novih programskih jezika, *debug* alata kao i drugih alata koji olakšavaju proces programskog razvoja. Ovom arhitekturom VSCode ima sposobnost da se konfiguriše u idealno razvojno okruženje nezavisno od programskog jezika, i samim tim uklanja potrebu instalacije drugih softverskih alata za pisanje izvornih kodova.

3. SPECIFIKACIJA SISTEMA

U ovom poglavlju je opisana specifikacija sistema za detekciju plagijarizma. Sistem pruža mogućnost lakog dodavanja komponenti koje imaju mogućnost pristupanja različitim skupovima podataka kao i izvršavanja različitih algoritama detekcije plagijarizma na datim skupovima dokumenata.

3.1. Servis za autentifikaciju i autorizaciju

Servis za autentifikaciju i autorizaciju obezbeđuje skup funkcionalnosti sa ciljem da određenim delovima sistema pristup imaju samo registrovani korisnici. Ovaj servis treba da obezbedi sledeće funkcionalnosti:

- Registracija korisnika u sistem
- Skladištenje informacija o korisnicima u sistemu
- Autentifikacija registrovanog korisnika
    - Izdavanje ključa za pristup određenim resursima sistema

Ostali servisi u sistemu pružaju određene funkcionalnosti koje se mogu koristiti samo uz upotrebu ključa za pristup datim funkcionalnostima.

3.2. Servis za detekciju plagijarizma

Servis za detekciju plagijarizma pruža mogućnost učitavanja sledećih komponenti prilikom izvršavanja: komponente za izvršavanje provere plagijarizma i komponente za pristup dokumentima.

Pored toga, ovaj servis nudi niz funkcionalnosti za rukovanje zadacima za proveru plagijarizma. Prilikom kreiranja zadatka za proveru plagijarizma, korisnik zadaje identifikator dokumenta koji se upoređuje sa referentnim dokumentima, identifikator konkretnog pristupa za proveru plagijarizma i identifikator referentnog skupa podataka. Zadaci se izvršavaju u asinhronom režimu. Korisnik u svakom trenutku može da proveri status zadataka koje je on kreirao. U slučaju da je neki zadatak izvršen, korisnik može da pregleda rezultate datog zadatka.

3.2.1. Komponente za detekciju plagijarizma

Komponente za detekciju plagijarizma moraju specificirati sa kojim tipovima dokumenata su kompatibilni kao i koji metapodaci su potrebni za upotrebu određene komponente za detekciju plagijarizma.

3.2.2. Komponente za pristup dokumentima

Svaka komponenta ovog tipa mora specificirati koji metapodaci su dostupni na svim dokumentima kojima pristupamo ovom komponentom. Ovo nam daje mogućnost da unapred znamo da li je određena komponenta za pristup dokumentima kompatibilna sa određenom komponentom za detekciju plagijarizma.

3.3. Servis za obradu dokumenata

Servis za obradu dokumenata pruža mogućnost otpremanja i skladištenja dokumenata i metadpodataka o datim dokumentima, kao i obradu datih dokumenata. Osnovna funkcionalnost ovog servisa je inicijalna obrada dokumenata prilikom koje se kreiraju osnovni metapodaci za dati dokument. Pored toga, ovaj servis daje mogućnost za učitavanje komponenti za dodatnu obradu svih dokumenata.

3.3.1. Komponente za obradu dokumenata

Komponente za obradu su zadužene da izvrše dodatnu obradu dokumenata i kao rezultat imaju podatke potrebne za proces detekcije plagijarizma.

3.4. Korisnički interfejs

Korisnički interfejs treba da omogući kontrole za registraciju i prijavu u sistem, kao i kontrole za rukovanje određenim funkcionalnostima sistema. U ove funkcionalnosti spadaju: pregled instaliranih komponenti za detekciju plagijarizma, pregled skupova dokumenata

odnosno komponenti za pristup skupovima dokumenata i rukovanje zadacima. Korisnici sistema mogu da kreiraju zadatke i da pregledaju rezultate datih zadataka upotrebom korisničkog interfejsa.

4. IMPLEMENTACIJA SISTEMA

U ovom poglavlju su predstavljeni implementacioni detalji svih delova i komponenti sistema. Opis delova i komponenti sistema podrazumeva pregled konceptualnih rešenja i upotrebljenih tehnologija.

Autentifikacija i autorizacija, servis za detekciju plagijarizma i servis za obradu dokumenata su implementirani pomoću Java 1.8 programskog jezika u formi veb aplikacija u Spring Boot radnom okviru.

4.1. Autentifikacija i autorizacija

Autentifikacija i autorizacija je implementirana pomoću Spring Security dodatka za Spring Boot radni okvir. Sistem ima dve različite uloge: administrator i nastavnik. Nastavnik ima mogućnost da otpremi i proveri da li je dokument plagijat dok administrator pored ovoga ima opciju da dodaje i briše dodatne komponente.

4.1.1. Model korisnika

Model korisnika obuhvata sledeće informacije: identfikator korisnika predstavljen pomoću UUID-a (eng. Universally Unique Identifier), email adresa, lozinka, ime, prezime i skup uloga koje korisnik ima u sistemu.

4.1.2. Registracija korisnika

Prilikom registracije, korisnik prosleđuje svoju email adresu, lozinku, ime i prezime servisu za autentifikaciju i autorizaciju, nakon čega servis dodeljuje odgovarajući identifikator datom korisniku i skladišti sve informacije ukoliko korisnik sa datom email adresom već ne postoji u sistemu.

Bitna bezbednosna stavka kod ovih sistema je da se lozinka nikako ne sme čuvati u izvornom obliku već je obavezno primeniti algoritme za heširanje i čuvati samo heširani oblik lozinke [5].

4.1.3. Prijava na sistem

Prilikom prijave na sistem, korisnik je dužan da prosledi svoju email adresu kao i lozinku koju je izabrao prilikom registracije na sistem. Nakon uspešne prijave korisniku se izdaje JWT token [6] (eng. JSON Web Token) sa kojim dobija mogućnost da pristupi ostalim resursima u sistemu što uključuje ovaj i ostale servise.

4.2. Dodavanje novih komponenti u runtime-u

Javin mehanizam za učitavanje klasa je deo izvršnog okruženja Jave, i ima mogućnost dinamičkog učitavanja Java klasa u Java virtuelnu mašinu [7].

Java biblioteke su tipično upakovane u JAR datoteke koje mogu da sadrže objekte različitih vrsta. Najbitnija vrsta objekta u JAR datoteci je Java klasa. Bitno je napomenuti da Javin mehanizam za učitavanje klasa ima mogućnost učitavanja klase sa određenim imenom samo jednom u toku jednog izvršavanja.

Kada se Javina virtuelna mašina pokrene tri klase za učitavanje klasa se koriste: *bootstrap class loader*, *extensions class loader* i *systems class loader*. *Bootstrap class loader* je zadužen za učitavanje Javinih ugrađenih biblioteka (eng. Core libraries) koji se nalaze u *<JAVA_HOME>/jre/lib* direktorijumu. *Extensions class loader* učitava dodatne biblioteke koje se nalaze u *<JAVA_HOME>/jre/lib/ext* direktorijumu. *System class loader* učitava klase koje se nalaze na classpath-u. Navedeni mehanizmi su implementirani u Javi i samim tim korisnici jezika mogu implementirati svoje mehanizma za učitavanje klasa. Jedan od tih mehanizama je *java.net.URLClassLoader* koji pruža mogućnost učitavanja klasa iz fajl sistema kao i JAR fajlova.

Za implementaciju učitavanje komponenti korišćen je navedeni *URLClassLoader*. U sistemu su definisani interfejsi koji omogućuju komponentama da se prilagode sistemu. Interfejsi su dati na listinzima 4.2.1., 4.2.2., 4.2.3 i 4.2.4.

```
public interface Plugin {
  PluginMetadata getPluginMetadata();
}
```

Listing 4.2.1. *Osnovni interfejs koji mora biti ispoštovan od strane svih komponenti.*

Klasa *PluginMetadata* sadrži sledeća polja: identifikator komponente, naziv komponente, opis komponente, listu naziva metapodataka koji su potrebni da bi data komponenta rukovala sa datim dokumentom, lista identifikatora komponenti koji se moraju učitati pre ove komponente.

```
public interface DetectorPluginFactory extends Plugin {
  DetectorPlugin createInstance();
}
```

Listing 4.2.2. *Interfejs za komponente koje vrše detekciju plagijarizma*

```
public interface DataAccessPlugin extends Plugin {
  Dataset loadDataset(DocumentMetadata candidateDocument);
}
```

Listing 4.2.3. *Interfejs za komponente koje vrše pristup dokumentima*

```
public interface ProcessingPlugin extends Plugin {
  String getResultFieldName();
  void process(DocumentMetadata documentMetadata,
Consumer<Object> storeValueForDocument);
}
```

Listing 4.2.4. *Interfejs za komponente koje vrše dodatnu obradu dokumenata*

4.3. Servis za detekciju plagijarizma

Servis za detekciju plagijarizma je veb aplikacija koja koordiniše između korisnika, komponenti za detekciju plagijarizma i komponenti za pristup dokumentima.

Pored toga sistem daje mogućnost korisnicima da dobiju osnovne informacije o svim komponentama koje su učitane u sistem. Prilikom zahteva za izlistavanje komponenti, sistem proverava da li postoje nove komponente u direktorijumu gde se nalaze instalirane komponente. U slučaju da postoje, sistem ih učitava i uključuje u rezultate zahteva.

Na osnovu dobijenih informacija korisnik može da napravi i pošalje zahtev za kreiranje zadatka, nakon čeka sistem skladišti dati zadatak i šalje odgovor korisniku da je zadatak uspešno kreiran. Sistem periodično proverava da li postoje zadaci koji trebaju da se izvrše, pokreće njihovo izvršavanje i skladišti rezultate.

4.3.1. Komponente za detekciju plagijarizma

Postojeća rešenja za detekciju plagijarizma se mogu adaptirati radi priključivanja u sistem. Ova implementacija se sastoji od dve adapter komponente koje adaptiraju dva postojeća rešenja za detekciju plagijarizma. Jedna komponenta pruža detekciju plagijarizma među tekstualnim dokumentima [8] dok druga radi sa Python izvornim kodovima [9].

4.3.2. Komponente za pristup dokumentima

Ova implementacija trenutno nudi jednu komponentu koja pruža pristup dokumentima koji se nalaze na servisu za obradu dokumenata.

4.4. Servis za obradu dokumenata

Servis za obradu dokumenata je implementiran u obliku veb aplikacije. Reprezentacije dokumenata su skladištene u fajl sistemu. Naziv sačuvanih datoteka je predstavljen jednim UUID-om. Metapodaci su skladišteni u MongoDB nerelacionoj bazi podataka. Osnovni metapodaci koji se čuvaju prilikom skladištenja dokumenta su originalni naziv dokumenta, skladišteni naziv dokumenta, tip datoteke, ekstenzija naziva datoteke. Dodatni metapodaci se generišu pomoću dodatnih komponenti za obradu.

4.4.1. Komponente za obradu dokumenata

Postojeća rešenja za obradu dokumenata se mogu adaptirati radi priključivanja u sistem. Ova implementacija se sastoji od dve adapter komponente koje adaptiraju dve postojeće implementacije obrade dokumenta. Jedna komponenta pruža ekstrakciju teksta iz PDF datoteka dok druga generiše MD5 *checksum*. Naziv datoteke u kojoj se nalazi ekstraktovani tekst, kao i MD5 checksum originalnog dokumenta se skladišti u skupu metapodataka datog dokumenta.

4.5. Korisnički interfejs

Korisnički interfejs je implementiran pomoću Jave u Vaadin 21 radnom okviru.

5. ZAKLJUČAK

Broj različitih tipova dokumenata raste velikom brzinom u različitim domenima u kojima intelektualna svojina igra bitnu ulogu. Cilj ovog rada je predstavljanje sistema koji ne ograničava korisnike na određeni predefinisani skup tipova dokumenata. U slučaju da se pojavi potreba za podršku novog tipa dokumenta, sistem se može lako proširiti adaptacijom postojećih rešenja za dati tip dokumenta ili implementacijom novog rešenja.

5.1. Dalji razvoj sistema

Sistemi ovog tipa imaju generalan problem performansi usled povećanog broja digitalnih dokumenata sa kojima oni raspolažu. Pretpostavimo da provera plagijarizma traje N sekundi po dokumentu u referentnom skupu. Za M dokumenata potrebno nam je $T = MN$ sekundi. Tradicionalno horizontalno skaliranje u ovom slučaju ne bi donelo do poboljšanja performansi prilikom provere individualnih dokumenata, već bi doprinelo slučajevima gde ima dosta zahteva od različitih korisnika. Da bismo skaliranjem dobili bolje performanse za individualne dokumente, skaliranje se mora implementirati tako da se referentni skup podeli u particije i da se ulazni dokument proverava sa svakom od particija.

6. LITERATURA

[1] East, J., 2006. The problem of plagiarism in academic culture. International Journal for Educational Integrity, 2(2).

[2] Jung, M., Mallering, S., Dalbhanjan, P., Chapman, P. and Kassen, C., 2016. Microservices on AWS. Amazon Web Services, Inc., New York, NY, USA, Tech. Rep.

[3] Belguidoum, M. and Dagnat, F., 2007. Dependency management in software component deployment. Electronic Notes in theoretical computer science, 182, pp.17-32.

[4] Microsoft (2016). Visual Studio Code. [online] Visualstudio.com. Dostupno na: https://code. visualstudio.com/. [Pristupljeno 6. 8. 2021]

[5] Sriramya, P. and Karthika, R.A., 2015. Providing password security by salted password hashing using bcrypt algorithm. ARPN journal of engineering and applied sciences, 10(13), pp.5551-5556.

[6] Jones, M., Campbell, B. and Mortimore, C., 2015. JSON Web Token (JWT) profile for OAuth 2.0 client authentication and authorization Grants. May-2015. [online]. Dostupno na: https://tools.ietf .org/html/rfc7523. [Pristupljeno 6. 8. 2021]

[7] Liang, S. and Bracha, G., 1998. Dynamic class loading in the Java virtual machine. Acm sigplan notices, 33(10), pp.36-44.

[8] GitHub. (n.d.). GitHub - JonathanReeve/text-matcher: A simple text reuse detection CLI tool. [online] Dostupno na: https://github.com/JonathanReeve/ text-matcher. [Pristupljeno 6. 8. 2021]

[9] GitHub. (n.d.). GitHub - fyrestone/pycode_similar: A simple plagiarism detection tool for python code. [online] Dostupno na: https://github.com/fyres tone/pycode_similar. [Pristupljeno 6. 8. 2021]

Kratka biografija:

Nikola Zeljković rođen je 30. januara 1996. godine u Novom Sadu, Republika Srbija. Godine 2015. upisao je Fakultet tehničkih nauka u Novom Sadu smer Softversko inženjerstvo i informacione tehnologije. 2019. godine diplomirao je na osnovnim studijama na Fakultetu tehničkih nauka, iste godine potom upisuje master studije na smeru Softversko inženjerstvo i informacione tehnologije. Kontakt: nikzeljkovic@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.85
DOI: https://doi.org/10.24867/15BE40Skipina

PREDIKCIJA RIZIKA I KLASIFIKACIJA OZBILJNOSTI SUDARA MOTORNIH VOZILA U NJUJORKU

TRAFFIC ACCIDENT RISK PREDICTION AND SEVERITY CLASSIFICATION IN THE CITY OF NEW YORK

Milica Škipina, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – *Svake godine u saobraćajnim nesrećama na putevima širom svijeta pogine 1,35 i bude povrijeđeno 20-50 miliona ljudi, što znači da svakog dana u prosjeku skoro 3.700 ljudi izgubi život u saobraćaju. Više od polovine poginulih su pješaci, motoristi ili biciklisti. U ovom radu će biti analizirani faktori koji direktno utiču na povećanje vjerovatnoće pojave sudara motornih vozila, a zatim predstavljena metodologija za predviđanje rizika kao i klasifikaciju nivoa ozbiljnosti sudara korištenjem klasifikacionih modela mašinskog učenja. Predloženi model objedinjuje podatke o sudarima, ulicama Njujorka, protoku saobraćaja na pojedinim dionicama i podatke o vremenu i može se koristiti za identifikaciju gdje i kada je rizik od nesreće značajno veći od prosjeka kako bi se preduzele radnje za smanjenje tog rizika. Rezultati modela za predikciju sudara dostižu tačnost od 70%, dok model za klasifikaciju ozbiljnosti sudara postiže makro-prosječni F1-skor od 0,56.*

Ključne reči: *saobraćajne nesreće, mašinsko učenje, klasifikacioni algoritmi, urbanističko planiranje*

Abstract – *Annually 1.35 million people worldwide die in road traffic, while 20-50 million get injured. That means that every day, almost 3,700 people are killed globally in crashes. More than half of those killed are pedestrians, motorcyclists, or cyclists. In this paper, we analyze the factors that directly affect the risk of motor vehicle crashes and propose the methodology for predicting traffic accidents and classification of collision severity. We were using various machine learning classification algorithms. We present a dataset obtained by extracting accident, road, traffic, and weather-related information from various data sources. The proposed model can identify the time and place when the risk of traffic accidents is higher so that risk can be reduced with proper actions. Experimental results show that the traffic accident prediction model can reach an accuracy of 70%, while the collision severity classification model achieves a macro-average F1-score of 0.56.*

Keywords: *traffic accidents, machine learning, classification algorithms, urban planning*

1. UVOD

Danas ljudi širom svijeta u cilju obavljanja svakodnevnih aktivnosti poput odlaska na posao ili u kupovinu za prelazak sa jedne na drugu lokaciju koriste različita prevozna sredstva. Posljedica toga je veliki broj sudara. Svake godine u saobraćajnim nesrećama u kojima učestvuju automobili, autobusi, motocikli, bicikli, kamioni ili pješaci na putevima pogine 1,35 i bude povrijeđeno 20-50 miliona ljudi širom svijeta. Više od polovine poginulih su pješaci, motoristi ili biciklisti. Na svaku osobu koja smrtno strada od posljedica povreda izazvanih u saobraćajnom udesu, na desetine ljudi koji su preživjeli za posljedicu imaju kratkoročni ili trajni invaliditet koji može rezultovati stalnim ograničenjima fizičkog funkcionisanja, psihosocijalnim posljedicama ili smanjenim kvalitetom života [1]. U Sjedinjenim Američkim državama (SAD), saobraćajne nesreće predstavljaju vodeći uzrok smrti osoba starosti od 19-54 godine i vodeći uzrok neprirodne smrti državljana SAD koji borave ili putuju u inostranstvo [2].

Saobraćajne nesreće ostavljaju i ekonomske posljedice. Procjenjuje se da će povrede sa ili bez tragičnog ishoda nastale u saobraćajnim udesima u periodu od 2015. do 2030. godine koštati svjetsku ekonomiju otprilike $1,8 triliona američkih dolara [3].

Uzimajući sve ovo u obzir, jako je bitno razmotriti sve moguće načine na koje bismo mogli spriječiti sudare ili reagovati efikasno u slučaju da do njih dođe. Pristup tačnim i ažuriranim informacijama o trenutnoj situaciji na putevima omogućava vozačima, pješacima i putnicima da donose bolje odluke prilikom kretanja u saobraćaju.

U ovom radu će biti analizirani faktori koji direktno utiču na povećanje vjerovatnoće pojave sudara motornih vozila, a zatim predstavljena metodologija za predviđanje kao i klasifikaciju nivoa sudara na sudare sa smrtnim ishodom i /ili ozbiljnim povredama i sudare u kojima je samo došlo do materijalne štete. Rješenje predstavljeno u ovom radu objedinjuje podatke iz različitih izvora: podatke vezane za sudare, razne informacije o ulicama Njujorka, informacije o protoku saobraćaja, kao i informacije o vremenskim prilikama za određene trenutke. Kako se problem predikcije sudara može posmatrati kao problem binarne klasifikacije gdje pozitivna klasa odgovara zabilježenom sudaru na određenoj lokaciji i u određenom trenutku, i za predikciju i za klasifikaciju ozbiljnosti sudara korišteni su različiti klasifikacioni algoritmi: *Random Forest, Extra Trees, LightGBM, XGBoost, CatBoost* i K najbližih komšija (*eng. K nearest neighbors*). Takođe je isproban

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bila dr Jelena Slivka, vanr. prof.

ansambl gdje su kombinovani različiti modeli u cilju poboljšanja performansi.

2. PRETHODNA RJEŠENJA

U [4] su korištena tri javno dostupna skupa podataka obezbijeđena od strane grada Montreala i vlade Kanade koji obuhvataju informacije o nesrećama (datum, vrijeme i lokacija), geometriji puteva i podatke o vremenskim prilikama koji su mjereni na svakih sat vremena na različitim meteorološkim stanicama.

Za kreiranje modela za predikciju sudara korištene su tehnike za analizu i obradu velikih skupova podataka (*eng. Big Data*). Problem neuravnoteženosti skupa podataka je rješen tako što su za negativne primjere izdvojili 0.1% nasumičnih uzoraka od 2,3 milijarde mogućih kombinacija vremenskih trenutaka i segmenata. Obučavali su tri modela bazirana na stablima: *Random Forest, Balanced Radnom Forest i XGBoost*. Kao najbolji se pokazao *Balanced Random Forest* koji tačno detektuje 85% sudara, uz preciznost od 28% i *false positive rate* od 13% na test skupu.

Faktori koji su identifikovani kao najvažniji prilikom predviđanja nesreća su: broj nesreća koje su se prethodnih godina dogodile na određenom segmentu puta, temperatura, dan u godini, sat i vidljivost na putu.

Cilj u [5] je pronalazak nove metode za izbor parametara za problem predikcije saobraćajnih nesreća u realnom vremenu. Poređena su dva načina za izbor parametara: *Random Forest* i *Frequent Pattern Tree* (FP). Podaci koji su korišteni obuhvataju informacije o saobraćajnim nesrećama koje su se dogodile na međudržavnom autoputu I-64 u Virdžiniji 2005. godine, podatke o vremenu, vidljivosti na putu, gustini saobraćaja, brzini kretanja i zauzetosti. Rezultati pokazuju da se FP pokazao kao bolji pri izboru parametara bez obzira na vrstu modela koja je korištena za predviđanje sudara, a u kombinaciji sa modelom *Bayesian network* može se predvidjeti 61,11% nesreća sa *false positive rate* od 38,16%. Najznačaniji atributi su osobine vezane za obim saobraćaja, dok su osobine vezane za brzinu rangirane mnogo niže.

Autori u [6] su razvijali model koji u realnom vremenu predviđa vjerovatnoću da će se dogoditi nesreća sa različitim nivoom ozbiljnosti i došli su do zaključka da različiti faktori utiču na pojavu nesreća različite ozbiljnosti. Korišteni su podaci prikupljeni na 29 milja dugom autoputu I-880 u Kaliforniji i podaci dobijeni sa pet meteoroloških stanica koje se nalaze na udaljenosti od oko 8km od autoputa. Za problem klasifikacije, posmatrana su tri nivoa ozbiljnosti sudara: nesreće sa smrtnim ishodom ili težim povredama, nesreće u kojima je bilo povrijeđenih bez smrtnih slučajeva i nesreće koje za posljedicu imaju samo materijalnu štetu (PDO). Rezultati pokazuju da su se PDO češće dešavale u uslovima kada je bio zagušen saobraćaj, sa promjenljivom brzinom i čestim promjenama trake, dok su se druge dvije grupe nesreća češće dešavale pri manjem zagušenju saobraćaja. Došli su do zaključka da velika brzina zajedno sa velikom razlikom u brzinama između susjednih traka povećava vjerovatnoću pojave fatalnih sudara.

3. SKUPOVI PODATAKA

Kako bi se što preciznije odredila vjerovatnoća pojave nesreće, kao i nivo ozbiljnosti nesreće, korištena su tri javno dostupna skupa podataka obezbijeđena od strane grada Njujorka (*NYC Open Data*) koji sadrže podatke o sudarima, informacije o pojedinačnim segmentima ulica i protoku saobraćaja na njima. Pored toga, pomoću API (*Application Programming Interface*) koj pruža *IBM Weather* [7] preuzete su informacije o vremenu. Svaki od skupova podataka će biti detaljnije opisan u nastavku.

3.1. Sudari motornih vozila

Motor Vehicle Collisions - Crashes [8] sadrži informacije svih policijskih izvještaja o sudarima motornih vozila u Njujorku koji se moraju popuniti u slučaju sudara gdje ima povrijeđenih ili smrtno stradalih osoba ili gdje postoji materijalna šteta u iznosu od najmanje $1.000. Dostupni su podaci od 01.07.2012. godine i ažuriraju se na dnevnom nivou. Svaki red u tabeli se odnosi na jedan sudar i sadrži jedinstven ID sudara, informacije o lokaciji, datumu i vremenu kada se desio sudar, naziv gradske oblasti (*eng. borough*), poštanski broj (*eng. ZIP code*), naziv ulice, najbliže raskrsnice, adresu, broj poginulih i povrijeđenih ljudi, kao i faktore koji su uticali na to da dođe do sudara i tipove vozila. Na slici 1 je prikazana mapa Njujorka sa crvenim tačkama koje označavaju lokacije na kojima je došlo do sudara.

3.2. LION

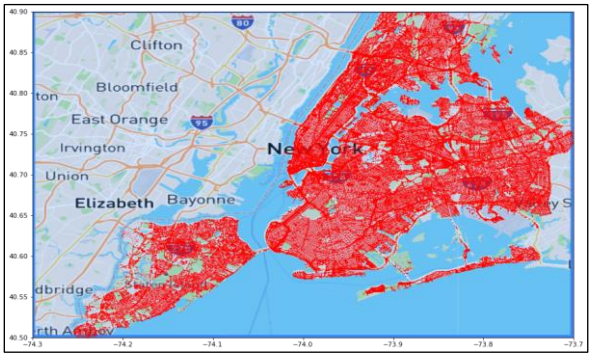
Linearno integrisana uređena mreža (*eng. Linear Integrated Ordered Network – LION*) [9] pomoću jedne linije predstavlja gradske ulice Njujorka i druge linearne karakteristike kao što su obale, željeznice i šetališta zajedno sa nazivima obilježja i opsegom adresa za svaki segment ulice koji se može adresirati. Za svaki segment ulice sadrži naziv ulice u kojoj se nalazi koji je iskorišten za popunjavanje nedostajućih vrijednosti kada je u [8] poznata lokacija, ali nije upisan naziv ulice. Pored naziva ulice, svaki segment je opisan informacijama poput smjera u kojem se odvija saobraćaj, širine, ograničenja brzine u miljama na sat, oznake da li segment pripada mreži ruta za bicikle ili kamione, prioritet za čišćenje snijega, broj saobraćajnih, parking i ukupnih traka na cesti, oznake da li je segment dostupan za pješake ili nije, kao i prostorne koordinate početka i kraja segmenta.

3.3. Protok saobraćaja

Traffic Volume Counts (2014 – 2019) [10] sadrži informacije o broju vozila koja su prošla određenim segmentom puta na svakih sat vremena određenog datuma. Podaci su dostupni samo za 422 datuma u periodu od 13.09.2014. do 24.11.2019. godine.

3.4. IBM Weather API

Informacije o vremenskim prilikama na svakih sat vremena za određene datume su preuzete pomoću *IBM Weather API*-ja. Dobavljene informacije uključuju temperaturu vazduha, subjektivni osjećaj, te količinu rose, vlažnosti, tip i jačinu vjetra kao i udara vjetra, vazdušni pritisak, količinu padavina, uslove (vedro, oblačno, kišovito, oluja,…), oznaku da li je dan ili noć i tip oblaka.



Slika 1. *Mapa Njujorka sa označenim lokacijama na kojima je došlo do sudara*

4. METODOLOGIJA

U ovom poglavlju će biti opisani izazovi sa kojima smo se susreli prilikom rješavanja opisanih problema i način na koji smo ih prevazilazili.

4.1. Integracija i pretprocesiranje podataka

Kako se opsezi datuma dostupnih podataka za [8] i [10] razlikuju, prvo su podaci o sudarima isfiltrirani i zadržani su samo oni koji su se dogodilli u periodu od 13.09.2014. do 24.11.2019. godine. Pošto lokacija predstavlja ključno obilježje za spajanje tri skupa podataka, odbačeni su svi uzorci kojima je ovaj atribut nedostajao. Takođe su izbačeni i redovi koji ni su sadržali informacije o broju poginulih i/ili povrijeđenih osoba jer se ovi atributi koriste prilikom labeliranja podataka za problem klasifikacije ozbiljnosti sudara.

Pošto različiti segmenti unutar jedne ulice mogu imati različite osobine (ograničenje brzine, broj saobraćajnih traka,…), za svaki red iz skupa podataka je na osnovu lokacije pronađen najbliži segment iz ulice u kojoj je došlo do sudara. Zatim su na osnovu njega dobavljene i ostale informacije vezane za taj segment. Spajanje podataka o sudarima sa informacijama o vremenskim prilikama je izvršeno na osnovu sata i datuma u kojem se desio sudar. Rezultujući skup podataka u sebi sadrži ukupno 1.025.439 zapisa o saobraćajnim nesrećama.

Prije spajanja sa skupom podataka o protoku saobraćaja, iz ovog skupa su usrednjene informacije o protoku sobraćaja za svaki od smjerova datog segmenta. Kako ovaj skup podataka sadrži informaije za samo 422 datuma, a postoje i razlike u oznakama segmenata, nakon spajanja sa ostalim skupovima podataka dobili smo novi koji u sebi sadrži 22.872 zapisa o saobraćajnim nesrećama. Kako je u prethodnim radovima zaključeno da ovo predstavlja jedno od najvažnijih obeležja, naše modele smo trenirali na dva odvojena skupa podataka: jedan koji uključivao podatke o protoku saobraćaja i drugi bez njih.

Iz dobijenih skupova podataka su izbačene sve kolone koje su imale veliki broj nedostajućih vrijednosti. Iako datum i vrijeme mogu sadržati informacije korisne za model, one ipak mogu biti neupotrebljive u standardnom formatu („DD-MM-YYYY HH:mm“), pa smo odlučili da u posebne kolone izdvojimo podatke o mjesecu, godini i satu, dok je dan transformisan u kolonu koja predstavlja dan u sedmici. Kako novi atributi prirodno predstavljaju ciklične podatke (podaci koji predstavljaju najudaljenije tačke u jednodimenzionalnoj ravni su u stvari najbliži, npr. 00h i 23h), transformisali smo ih u dvije dimenzije pomoću sinusne i kosinusne transformacije. Takođe je lokacija koja u našem skupu podataka predstavljena pomoću geografske širine i dužine mapirana na *x, y i z* koordinade.

Za kategorička obilježja je korišten *one-hot encoding*, dok su za kolonu koja predstavlja uslove na putu prethodno grupisane vrijednosti u tri grupe: normalni uslovi na puti ili oblačno vrijeme, uslovi sa slabijom vidljivošću i uslovi opasni za vožnju.

4.2. Labeliranje i neuravnoteženost skupa podataka

Za problem klasifikacije ozbiljnosti sudara, svaki od uzoraka je pridružen jednoj od klasa: 1 – ukoliko je u nesreći bilo povrijeđenih ili poginulih osoba i 0 u suprotnom. Kako je broj uzoraka koji pripadaju klasi 1 znatno manji od onih koji pripadaju klasi 0, prije treniranja je izvršen *resampling* podataka, odnosno, prvo je nasumično izbačen određeni broj uzoraka koji pripadaju klasi 0, a zatim je pomoću SMOTE (*Synthetic Minority Oversampling TEchnique*) algoritma povećan broj uzoraka klase 1.

Kod problema predikcije sudara, pozitivnoj klasi odgovaraju svi uzorci iz skupa podataka. Da bi se mogao obučiti model, prethodno je bilo potrebno izgenerisati negativne primjere. Za svaki uzorak iz skupa podataka, generisan je po jedan negativni primjer koji je imao isti datum dok su ulica i vrijeme birani nasumično pod uslovom da se u datoj ulici istog dana nije desio nijedan sudar.

4.3. Obučavanje modela

Prije obučavanja, skup podataka je podijeljen na trening i test skupove u odnosu 90% za trening i 10% za test skup. Pošto oba posmatrana problema predstavljaju problem binarne klasifikacije, obučavani su modeli zasnovani na stablu (*Random Forest, Extra Trees, LightGBM, XGBoost* i *CatBoost*). Zbog načina na koji je izvršen resampling podataka, pored njih je isproban i model K najbližih komšija.

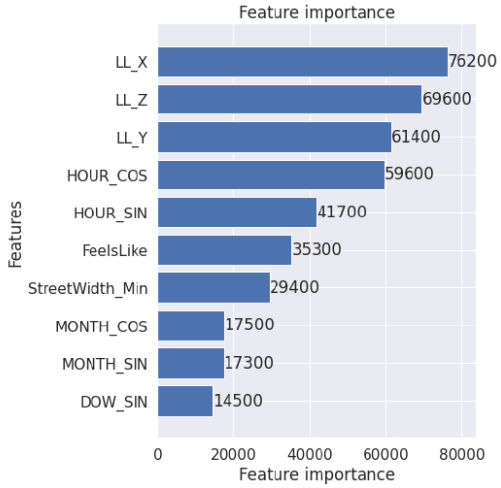
5. Evaluacija rješenja i rezultati

Za evaluaciju modela predikcije korištene su metrike: preciznost (*eng. precision*), tačnost (*eng. accuracy*), *recall* i *F1*-skor, dok su za problem klasifikacije ozbiljnosti sudara korištene makro-prosječne vrijednosti ovih metrika (osim tačnosti). U tabeli 1 su prikazani rezultati makro prosječnog F1-skora svih modela obučenih na skupu podataka bez informacija o protoku saobraćaja.

Tabela 1. *F1-skor modela obučenih na skupu podataka bez informacija o protoku saobraćaja*

Model	Predikcija sudara	Klasifikacija ozbiljnosti
Extra Trees	0,64	0,53
LightGBM	0,68	**0,56**
XGBoost	0,69	**0,56**
CatBoost	**0,70**	0,55
KNN	0,46	0,52
Ensemble	0,69	**0,56**

Kao najbolji model za predikciju sudara se pokazao *CatBoost* koji dostiže F1-skor od 0,70, dok su kod klasifikacije ozbiljnosti sudara sličan F1-skor od 0,56 ostvarila dva modela – *LightGBM* i *XGBoost*. Dodatno je isproban ansambl dva najbolja modela za svaki od problema, ali nije došlo do poboljšanja performansi. Analizom rezultata je utvrđeno da modeli bazirani na stablu postižu približno slične performanse. KNN se pokazao kao znatno lošiji za problem predikcije sudara, dok se manja razlika primjećuje za model koji je istreniran za klasifikaciju ozbiljnosti sudara. Razlog za ovo je vjerovatno SMOTE tehnika za generisanje novih primjera manjinske klase koji u osnovi koristi KNN. Na slici 2 je dat prikaz top 10 najznačajnijih atributa *LightGBM* modela za klasifikaciju ozbiljnosti sudara, gdje se može vidjeti da najvažnije atribute prilikom predikcije predstavljaju lokacija i vrijeme sudara, zatim subjektivni osjećaj, širina ulice, mjesec i dan u sedmici.



Slika 2. *Mapa Njujorka sa označenim lokacijama na kojima je došlo do sudara*

Kako bismo povećali performanse, za problem klasifikacije ozbiljnosti sudara je istreniran model nad skupom podataka koji sadrži informacije o protoku saobraćaja. Kao najbolji model se pokazao *Extra Trees* koji dostiže makro-prosječan F1-skor od 0,69. Iako je ovo značajno poboljšanje, rezultati nisu direktno uporedivi jer ovaj skup podataka sadrži mnogo manje uzoraka u poređenju sa prvim.

5. ZAKLJUČAK

U ovom radu je predstavljeno rješenje za predviđanje lokacije i trenutka u kojem je povećan rizik pojave sudara, kao i klasifikacija ozbiljnosti sudara motornih vozila u Njujorku. Objedinjeni su skupovi iz različitih izvora podataka nad kojim su zatim trenirani različiti klasifikacioni modeli. Nakon rješenja problema neuravnoteženosti skupa podataka, rezultati pokazuju da najbolje performanse prilikom klasifikacije ozbiljnosti sudara postižu modeli *LightGBM* i *XGBoost* sa F1-skorom od 0,56. Za problem predikcije sudara, prvo je bilo neophodno generisati podatke koji se odnose na lokaciju i trenutak u kojem nije došlo do sudara. Nakon treniranja i evaluacije modela, utvrđeno je da najbolje performanse postiže *CatBoost* sa F1-skorom od 0,70. Dodavanje podataka vezanih za protok saobraćaja značajno poboljšava rezultate modela za klasifikaciju nivoa ozbiljnosti sudara, ali rezultati nisu robusni jer su modeli trenirani na znatno manjem skupu podataka.

Predviđanjem težine nesreće moglo bi se efikasnije reagovati u hitnim slučajevima. Na pojavu sudara utiču i drugi faktori kao što je ponašanje vozača, što je teško detektovati kako bi se moglo iskoristiti kao jedan od dodatnih atributa, ali bi modeli mogli biti poboljšani upotrebom podataka vezanih za gustinu naseljenosti određenog mjesta ili pojedinačnih dijelova grada.

6. LITERATURA

[1] Peden, Margaret. (2004). World Report on Road Traffic Injury Prevention.

[2] Centers for Disease Control and Prevention (CDC), National Center for Injury Prevention and Control (NCIPC). Web-based Injury Statistics Query and Reporting System (WISQARS). Available from URL: http://www.cdc.gov/injury/wisqars

[3] Chen, Simiao & Kuhn, Michael & Prettner, Klaus & Bloom, David.(2019). The global macroeconomic burden of road injuries: estimates and projections for 166 countries. The Lancet Planetary Health. 3. e390-e398. 10.1016/S2542-5196(19)30170-6.

[4] A. Hébert, T. Guédon, T. Glatard and B. Jaumard, "High Resolution Road Vehicle Collision Prediction for the City of Montreal," 2019 IEEE International Conference on Big Data (Big Data), 2019, pp. 1804-1813.

[5] L. Lin, Q. Wang, and A. W. Sadek, "A novel variable selection method based on frequent pattern tree for real-time traffic accident risk prediction," Transportation Research Part C: Emerging Technologies, vol. 55, pp. 444 – 459, 2015

[6] Xu C, Tarko AP, Wang W, Liu P. Predicting crash likelihood and severity on freeways with real-time loop detector data. Accid Anal Prev. 2013 Aug;57:30-9. doi: 10.1016/j.aap.2013.03.035. Epub 2013 Apr 6. PMID: 23628940.

[7] https://www.ibm.com/weather

[8] https://data.cityofnewyork.us/browse?Data-Collection_Data-Collection=Motor+Vehicle+Collisions

[9] https://www1.nyc.gov/site/planning/data-maps/open-data/dwn-lion.page

[10] https://data.cityofnewyork.us/Transportation/Traffic-Volume-Counts-2014-2019-/ertz-hr4r

Kratka biografija:

Milica Škipina rođena je 1997. godine u Srbinju. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva odbranila je 2021. godine.

kontakt: skipinamilica@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/15BE41Bucan

MAŠINSKO UČENJE NA IVICI UPOTREBOM NVIDIA JETSON TX2 UREĐAJA

MACHINE LEARNING ON THE EDGE USING NVIDIA JETSON TX2 DEVICE

Dušan Bućan, Fakultet tehničkih nauka, Novi Sad

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu opisana je primena konvolucionih neuronskih mreža u oblasti računarske vizije, kao i tehnike obrade toka podataka i arhitektura sistema za demografsku analitiku u realnom vremenu upotrebom mašinskog učenja na ivici. Opisan sistem je implementiran i izloženi su mogući pravci unapređenja i proširenja sistema.*

Ključne reči: *mašinsko učenje na ivici, računarska vizija, obrada toka podataka, demografska analitika u realnom vremenu*

Abstract – *In this work we present use of convolution neural networks in computer vision, stream processing techniques and an architecture of system for real-time demography analytics using machine learning on the edge. Implementation of described system and future improve-ments of system.*

Keywords: *machine learning on the edge, computer vision, stream processing, real-time demography analytics*

1. UVOD

Svakodnevno se kreira velika količina video materijala sa kamera postavljenih na javnim i privatnim mestima u gradu. Obrada ovih podataka bi mogla da unapredi kvalitet života i bezbednost građana, što predstavlja i jedan od motiva za razvoj sistema koji koriste mašinsko učenje na ivici. Mašinsko učenje na ivici (engl. *Machine learning on the edge*) [1] predstavlja novu oblast primene mašinskog učenja koja ima za cilj široku primenu modela mašinskog učenja na ugrađenim sistemima (engl. *embedded system*) [2]. Uređaji na ivici obrađuju ulazne podatke upotrebom modela mašinskog učenja i šalju procesirane podatke na centralni server, čime se prenosi manje podataka uz veću bezbednost.

Prenos manjeg obima procesiranih podataka čini sisteme sa mašinskim učenjem na ivici pogodnim za obradu velike količine podataka u realnom vremenu, zbog čega ovi sistemi često imaju arhitekturu zasnovanu na tokovima podataka. Kod arhitektura zasnovane na tokovima podataka podaci sa ugrađenih uređaja se upisuju u tok podataka u realnom vremenu nad kojim se zatim radi analiza i prikaz rezultata obrade u realnom vremenu.

U ovom radu izložena je ideja demografske analitike u realnom vremenu na osnovu video snimka upotrebom mašinskog učenja na ivici. Sistem se sastoji iz:

- Podsistema za obradu izvornih podakata na ivici
- Podsistema kontrolne jedinice
- *MLOps* [3] podsistema.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji je mentor bio dr Dušan Gajić, vanr. prof.

Podsistem za obradu izvornih podataka na ivici je sistem implementiran na *Nvidia Jetson TX2* uređaju [4], u nastavku *Jetson*. Ovaj podsistem na ulazu prima video snimak nad kojim obavlja procesiranje, odnosno radi demografsku analizu ljudi.

Podsistem kontrolne jedinice ima za cilj obradu tokova podataka koji sadrže informacije o demografskim odlikama ljudi u realnom vremenu, prikaz rezultata obrade tokova podataka kao i upravljanje ugrađenim sistemima na ivici.

Cilj postojanja *MLOps* podsistema i upotrebe *MLOps* praksi je brži razvoj sistema koji koriste modele mašinskog učenja, brža adaptacija na promenu zahteva, jednostavnije beleženje eksperimenata i poređenje modela mašinskog učenje, kao i uvođenje istih u produkciju.

2. PRETHODNA REŠENJA

U ovom poglavlju opisan je jedan od razmatranih sistema koji ima sličnu arhitekturu kao sistem mašinskog učenja na ivici za demografsku analitiku stanovništva. Sistemi za obradu velikih skupova podataka na ivici zasnovanom na klasteru ugrađenih sistema.

Rad [5] predstavlja ideju obrade velikih skupova podataka na ivici u približno realnom vremenu. Predloženo rešenje implementira sistem koji se sastoji od klastera *Raspberry Pi* [5] uređaja. Prednost predloženog rešenja u odnosu na tradicionalne tehnike obrade velikih skupova podataka je to što se podaci obrađuju na *Raspberry Pi* uređajima bliže mestu prikupljanja čime se obrada ubrzava. Obrada podataka u sistemu izloženom u radu [5] je realizovana upotrebom *Spark* [5] alata i *HDFS* [5] tehnologije. U poređenju sa radom [5] naš sistem koristi *Kafka* klaster [6] umesto *HDFS* što ga čini pogodnijim za obradu velikog skupa podataka u realnom vremenu kad je *Spark* alat zamenjen *Flink* [7] alatom za obradu toka podataka. Sistem izložen u radu [5] koristi *Docker* [5] i *Docker Swarm* [5] kako bi sistem učinio otpornijim na otkaze. Upotreba navedenih tehnologija uparena sa *Prometheus* i *Grafana* alatima [5] doprinosi lakšem nadgledanju i upravljanju celokupnim sistemom što bi moglo predstavljati pravac daljeg unapređenja našeg sistema. Naš sistem trenutno koristi *Docker* samo kao izvršno okruženje alata za procesiranje toka podataka i prikaz rezultata analitike toka podataka ali ne i za praćenje stanja ugrađenih uređaja.

3. SPECIFIKACIJA I IMPLEMENTACIJA REŠENJA

Ovo poglavlje posvećeno je korišćenim alatima (poglavlje 3.1), specifikaciji rešenja i arhitekture sistema (poglavlje 3.2), i skupovima podataka korišćenim za treniranje i testiranje modela (poglavlje 3.3).

3.1 Korišćeni alati

Pri implementaciji podsistema za obradu izvornih podataka na ivici korišćen je *Python* programski jezik, *Python OpenCV* biblioteka [8] za obradu slike kao i *Python* biblioteke za interakciju sa *Kafka message broker*-om [6], *MinIO* [9] skladištem podataka i *TensorRT* [10] okruženjem za izvršavanje modela. Modeli za detekciju ljudi i lica, klasifikaciju pola kreirani su, trenirani i testirani upotrebom *Tensorflow* biblioteke [11].

Pri implementaciji podsistema kontrolne jedinice za potrebe obrade toka podataka korišćen je alat *Flink* i programski jezik *Java.* Centralnu komponentu podsistema čini *Kafka message broker* koja pruža podršku za upis, čuvanje i čitanje poruka. *Za* smeštanje rezultata analize toka podataka korišćeni su *Kafka connect* [6] softverski alat i *PostgreSQL* [12] baza podataka. Vizualizacija uskladištenih rezultata obrade toka podataka implementirana je upotrebom *Superset* alata [13] i *SQL*-a. *MLOps* podsistem implementiran je upotrebom *MLFlow* alata [14], *PostgreSQL* relacione baze podataka, *MinIO* objektnog skladišta podataka i *Tensorflow Docker* kontejnera za treniranje modela. Infrastruktura *MLOps* podsistema i infrastruktura kontrolne jedinice su implementirane upotrebom *Docker* tehnologije, tačnije alati su korišćeni kao *Docker* kontejneri.

3.2 Arhitektura rešenja

Rešenje se sastoji iz tri odvojena podsistema – podsistema za obradu izvornih podataka na ivici (poglavlje 0), podsistema kontrolne jedinice (poglavlje 0) i *MLOps* podsistema (poglavlje 0). Arhitektura celokupnog sistema predstavljena je grafički na Slika 1.

3.2.1 Podsistem za obradu izvornih podataka na ivici

Komponente koje čine podsistem za obradu izvornih podataka su:

- Komponente za učitavanje videa
- Komponente za čuvanje videa
- Komponente za slanje poruka
- Komponente za detekciju ljudi
- Komponente za detekciju lica
- Komponente za klasifikaciju pola
- *Pipeline* komponenta

Komponente za učitavanje videa pružaju funkcionalnost učitavanja videa frejm po frejm sa različitih izvora podataka. Trenutna implementacija podržava učitavanje videa sa:

- *Web* kamere
- Ugrađene kamere na *Jetson* uređaju
- Lokalnog fajl sistema

Komponente za čuvanje videa pružaju funkcionalnost čuvanja videa na različitim tipovima skladišta. Trenutna implementacija podržava čuvanje videa na:

- Lokalnom fajl sistemu
- *MinIO* objektnom skladištu podataka

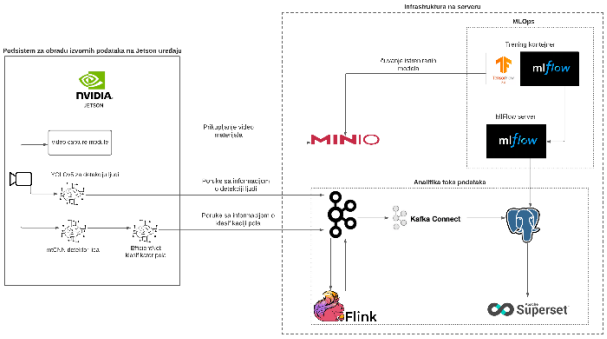
Komponente za slanje poruka pružaju funkcionalnosti kreiranja poruka na osnovu ulaznih, obrađenih podataka kao i slanja istih na *Kafka message broker.*

Komponente za detekciju ljudi nad ulaznim frejmovima detektuju ljude, tj. njihove granične okvire. Postoje dve implementacije komponente za detekciju ljudi i obe u osnovi koriste pretreniran *YOLOv5m* detektor [15]. Jedna komponenta za detekciju ljudi je prilagođena efikasnijem izvršavnju na *Jetson* uređaju, dok se druga koristi u svrhe testiranja na lokalnom računaru. Komponenta za detekciju ljudi na *Jetson*-u je konvertovana u *TensorRT* format, dok je komponenta za upotrebu na lokalnom računaru kovertovana u *Tensorflow* format iz *PyTorch* [16] formata, koji je izvorni format detektora.

Komponente za detekciju lica nad ulaznim frejmovima detektuju lica, tj. granične okvire lica. Postoje dve implementacije komponente za detekciju lica i obe u osnovi koriste pretreniran *mtCNN* detektor [17]. Jedna komponenta je prilagođena efikasnijem izvršavanju na *Jetson* uređaju, dok se druga koristi u svrhe testiranja na lokalnom računaru. Komponenta za detekciju lica na *Jetson* uređaju konvertovana je u *TensorRT* format, dok je komponenta za upotrebu na lokalnom računaru konvertovana u *Tensorflow* format iz *PyTorch* formata.

Komponente za klasifikaciju pola na ulazu primaju izdvojene regione od interesa koje je komponenta za detekciju lica prepoznala kao lice a na izlazu kreiraju skup predikcija pola za svaki od ulaza. Postoje dve implementacije komponente za detekciju lica i obe u osnovi koriste ručno kreiranu neuronsku mrežu. Ručno kreiranu mrežu čine *EfficientNet B7* [18] konvolutivna neuronska mreža bez potpuno povezanih slojeva sa dodatim potpuno povezanim slojem od 128 neurona i izlaznim slojem koji sadrži jedan neuron.

Pipeline komponenta se sastoji od drugih komponenti i ima za cilj koordinaciju drugih komponenti.



Slika 1. Arhitektura sistema

3.2.2 Podsistem kontrolne jedinice

Podsistem kontrolne jedinice se sastoji iz više komponenti:

- Kontrolne table na kojoj je moguće pratiti demografsku analitiku u realnom vremenu
- Modula za analitiku u realnom vremenu (engl. *real-time analytics*)
- Modula za kontrolu ugrađenih sistema na ivici

Kontrolna tabla je implementirana uz pomoć *Apache Superset* alata, odnosno koristi se *Apache Superset Docker* kontejner. Kontrolna tabla vizualizuje uskladištene podatke iz *PostgreSQL* baze podataka. Podaci su prikazani kao dva grafika od kojih jedan prikazuje prosečan broj detektovanih ljudi u minutu, a drugi prosečan broj detektovanih muškaraca i žena u minutu.

Modul za analitiku u realnom vremenu predstavlja aplikaciju za procesiranje toka podataka u realnom vremenu. *Apacke Flink* alat je korišćen u implementaciji modula zbog jednostavnosti upotrebe, integracije sa *Docker* alatom, lakoće skaliranja i ugrađenih strategija oporavka od

greške. *Apache Flink* predstavlja distribuirano izvršno okruženje za aplikacije obrade toka podataka. Osnovnu arhitekturu *Apache Flink* klastera čine dva čvora, jedan *JobManager* [7] i jedan *TaskManager* [7].
Uloga *JobManager*-a je organizacija i nadgledanje izvršavanja aplikacije za obradu toka podataka, dok je uloga *TaskManager*-a izvršavanje delova aplikacije. Pri implementaciji aplikacije za obradu toka podataka korišćena je *Flink API* biblioteka za *Java* programski jezik. U sistemu su implementirane dve aplikacije za obradu toka podataka:

- aplikacija za obradu toka podataka detekcije ljudi
- aplikacija za obradu toka podataka klasifikacije pola ljudi

Aplikacija za obardu toka podataka detekcije ljudi na ulazu prima poruke sa *peopleDetections Topic*-a iz *Kafka message broker*-a, prikazano na Slika 1. Aplikacija grupiše podatke koji pristižu u realnom vremenu u vremenske intervale od po minut. Nad svakim vremenskim intervalom određuje prosečan broj ljudi u tom vremenskom intervalu. Izlaz aplikacije za obradu toka podataka detekcije ljudi su poruke sa informacijma o prosečnom broju detektovanih ljudi kao i početak i kraj vremenskog intervala u kojem je prosečan broj ljudi računat. Izlazne poruke se upisuju u *peopleAvgCount Topic Kafka message broker*-u.
Aplikacija za obradu toka podataka klasifikacije pola ljudi radi po istom principu, jedina razlika je u čitanju podataka iz drugog toka podataka, *genderClassifaction Topic* iz *Kafka message broker*-a, kao i pisanje u drugi izlazni tok podataka, *avgGenderClassifaction*. Implementirane aplikacije za obradu tokova podataka se postavljaju na *JobManager* čvor *Flink* klastera i nakon čega počinje njihovo izvršavanje.
Upotrebom *Kafka Connect* alata poruke iz *Kafka Topic*-a, *peopleAvgCount* i *avgGenderClassifaction* se skladište u *PostgreSQL* bazi podataka, kao na Slika 1.
Modul za kontrolu ugrađenih sistema na ivici je implementiran u vidu *Python* skripte. U trenutnoj verziji sistema moguće je konfigurisati ugrađeni sistem samo pre početka njegove upotrebe, odnosno moguće je odabrati jednu od četiri opcije:

- Prikupljanje video materijala sa kamere
- Detekcija ljudi na videu
- Klasifikacija ljudi na videu
- Detekcija ljudi i klasifikacija ljudi na videu

3.2.3 *MLOps* podsistem

MLOps podsistem je implementiran upotrebom *MlFlow* platforme kao na Slika 1. *MLOps* podsistem čine:

- repozitorijum modela mašinskog učenja
- baza podataka sa informacijama o treniranju modela
- baza podataka sa prikupljenim video materijalom za treniranje modela
- *MlFlow* server koji predstavlja kontrolnu tablu za prikaz informacija o treniranim modelima

Repozitorijum modela mašinskog učenja i baza podataka sa video materijalima za treniranje modela mašinskog učenja su realizovane kao objektne baze upotrebom *MinIO* tehnologije. Baza podataka koja sadrži informacije o treniranju modela je realizovana kao relaciona baza podataka upotrebom *PostgresSQL* baze podataka. *MlFlow* platforma poseduje *Python API* biblioteku koju je potrebno koristiti pri interakciji sa *MLFlow* serverom. Treniranje modela mašinskog učenja se najčešće odvija unutar kontejnera za trening, obično *Tensorflow* ili *PyTorch* kontejner. Za vreme treninga potrebno je da trening kontejner šalje informacije o konfiguraciji treninga kao i informacije generisane tokom treninga upotrebom *MLFlow Python API* biblioteke. Implementirano je slanje *F1 score* [19] mere kao i slanje preciznosti, tačnosti nad validacionim skupom po epohama, vreme trajanja svake od epoha.

3.3 Skupovi podataka

Korišćen *YOLOv5m* detektor ljudi pretreniran je na *COCO* [20] skupu podataka od strane autora detektora. Detektor lica, *mtCNN*, je pretreniran na *WIDER FACE* [21] i *CelebA* [22] skupovima podataka od strane autora detektora. Za treniranje i testiranje modela za prepoznavanje pola ljudi korišćen je *UTKFace* [23] skup podataka, dok je osnova modela, *EfficientNet B7* bez potpuno povezanih slojeva, pretrenirana na *ImageNet* [24] skupu podataka. *COCO*, *WIDER FACE*, *CelebA*, *ImageNet* predstavljaju široko korišćene i poznate skupove podataka pa je njihov pregled izostavljen. U narednom poglavlju je detaljnije opisan *UTKFace* skup podataka.

3.3.1 Skup podataka za prepoznavanje pola ljudi

UTKFace skup podataka je javno dostupan i sadrži preko dvadeset tri hiljade šesto sedamdeset osam (23678) uzoraka sa anotacijama o polu, godištu i etničkoj pripadnosti. Uzorci unutar skupa podataka su raznovrsni, odnosno postoje slike lica različitog osvetljenja, rezolucije, ugla slikanja lica, ljudi različitih starnosnih grupa. Skup podataka pokriva starostni raspon ljudi od nula (0) godina do sto šesnaest (116). Skup podataka je namenjen za raznovrsnu upotrebu, neke od primena su prepoznavanje lica, prepoznavanje pola, procena godina ljudi. Od ukupnog broja uzoraka, 23678, za trening je korišćeno 17048 uzoraka koji su nasumično odabrani, dok je za validacioni skup nasumično odabrano 1894 uzorka, ostalih 4735 uzoraka čine test skup.

4. EVALUACIJA REŠENJA I REZULTATI

U ovom poglavlju biće reči samo o evaluciji podsistema za obradu izvornih podataka na ivici. Evaluacija podsistema kontrolne jedinice je izostavljena jer su pri implementaciji korišćeni najbolji alati za domenski problem. Poznavajući činjenicu da je maksimalan broj poruka koje je moguće upisati u *Kafka messagr broker* u jednoj sekundi dva miliona poruka, dok podsistem kontrolne jedinice upisuje svega dve poruke po sekundi može se zaključiiti da podsistem kontolne jedinice koristi alate daleko ispod maksimalne granice, što čini evaluaciju ovog podsistema suvišnom. Kao što je opisano u poglavlju 0 podsistem za obradu izvornih podataka na ivici poseduje više komponenti koje sadrže modele mašinskog učenja koji su evaluirani.

4.2. Evaluacija i rezultati *YOLOv5m* detektora ljudi

Za evaluaciju korišćenog *YOLOv5m* detektora odabrana je *mAP* [19] metrika. Vrednost za *mAP* metriku sa 0.5 graničnom vrednošću *IoU* nad *COCO* validacionim skupom

je izračunata od strane autora *YOLOv5m* detektora i iznosi 63.1%. Dodatna evaluacija *YOLOv5m* detektora je urađena nad ručno kreiranim skupom podataka koji se sastoji od slika prikupljenih upotrebom ranije pomenute komponente za čuvanje videa. Ručno kreirani skup podataka sadrži 21 sliku i ukupno 71 stvarnih graničnih regiona ljudi. Izračunata vrednost za *mAP* metriku sa 0.5 graničnom vrednošću *IoU* nad ručno kreiranim skupom podataka je 0.845%. Zbog visoke vrednosti *mAP* pri detekciji ljudi dodatni trening *YOLOv5m* detektor na ručno kreiranim skupom podataka je izostavljen, jer kreiranje trening skupa zahteva dosta resursa a ne bi značajno unapredilo performanse sistema.

4.3. Evaluacija i rezultati *mtCNN* detektora lica

Za evaluaciju korišćenog *mtCNN* detektora korišćena je *AP* metrika i funkcija preciznosti po odzivu, opisana ranije na *WIDER FACE* skupu podataka od strane autora detektora. Postignuti rezultati svrstavaju korišćeni *mtCNN* detektor među najbolje detektore lica testirane na *WIDER FACE* test skupu. *WIDER FACE* skup podataka poseduje veliki diverzitet, pa prikupljeni video materijal sa kamere *Jetson*-a na kojem su zabeležena lica ljudi čini njegov podskup. Iz ovoga je moguće zaključiti da dodatni trening *mtCNN* detektora nije potreban.

4.4 Evaluacija i rezultati testiranja klasifikatora pola

Model kreiran za klasifikaciju pola je evaluiran upotrebom *F1 score* metrike na test delu *UTKFace* skupa podataka. Test deo *UTKFace* skupa se sastoji od 4735 uzoraka nasumično odabranih pre treninga modela. Model postiže 0.855 *F1 score* pri predviđanju ženskog pola i 0.856 *F1 score* pri predviđanju muškog pola na kreiranom test skupu.

5. ZAKLJUČAK

U ovom radu predstavljen je sistem za demografsku analitiku u realnom vremenu na osnovu video snimka upotrebom mašinskog učenja na ivici. Jedna od mogućih primena je bolje određivanje ciljne grupe kupaca u prodavnicama. Ako bi se posmatrao broj i osobine ljudi koji prilaze određenom artiklu moguće bi bilo odrediti ciljnu grupu za taj artikal. U malim prodavnicama ovo je moguće realizovati bez upotrebe ovakvog sistema, ali u slučaju velikih prodavnica ili za vreme intenzivnih perioda kupovine sistem bi zasigurno nadmašio učinak prodavaca i u tom slučaju bi bio doprineo razvoju prodavnice.

Moguća unapređenja implementacije sistema opisanog u poglavlju 3.2 Arhitektura rešenjasu poboljšanje performansi podsistema za obradu izvornih podataka na ivici kao razvoj modula za inteligenciju mnoštva (engl. *Swarm intelligence*) [25]. Navedeni podsistem se oslanjaju na konvolutivne neuronske mreže pri određivanju pola osoba čije bi se performanse mogle poboljšati povećanjem obima skupa podataka za trening. Upotreba ansambla klasifikatora pola bi takođe doprinela porastu performansi sistema. Razvoj modula za inteligenciju mnoštva omogućio bi koordinisanu i istovremenu upotrebu više ugrađenih uređaja i samim tim unapredio i proširio primenu celog sistema.

6. LITERATURA

[1] Chang, Zheng, et al. "Learn to cache: Machine learning for network edge caching in the big data era." *IEEE Wireless Communications* 25.3 (2018): 28-35.
[2] https://en.wikipedia.org/wiki/Embedded_system (pristupljeno 15.08.2021.)
[3] https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning (pristupljeno 15.08.2021.)
[4] https://developer.nvidia.com/embedded/jetson-tx2 (pristupljeno 16.08.2021.)
[5] Scolati, Remo, et al. "A Containerized Big Data Streaming Architecture for Edge Cloud Computing on Clustered Single-board Devices." *Closer*. 2019.
[6] https://kafka.apache.org/documentation/ (pristupljeno 19.08.2021.)
[7] https://flink.apache.org/, poslednji pristup: 23.08.2021.
[8] https://docs.opencv.org/4.5.2/d6/d00/tu-torial_py_root.html (pristupljeno 24.08.2021.)
[9] https://docs.min.io/docs/minio-quickstart-guide.html (pristupljeno 23.08.2021.)
[10] https://docs.nvidia.com/deeplearning/tensorrt/develo-per-guide/index.html (pristupljeno 25.08.2021.)
[11] https://www.tensorflow.org (pristupljeno 25.08.2021.)
[12] https://www.postgresql.org/ (pristupljeno 23.08.2021.)
[13] https://superset.apache.org/ (pristupljeno 22.08.2021.)
[14] https://mlflow.org/docs/latest/index.html, (pristupljeno 23.08.2021.)
[15] https://github.com/ultralytics/yolov5 (pristupljeno 16.08.2021.
[16] https://pytorch.org/ (pristupljeno 24.08.2021.)
[17] Zhang, Kaipeng, et al. "Joint face detection and alignment using multitask cascaded convolutional networks." *IEEE Signal Processing Letters* 23.10 (2016): 1499-1503.
[18] Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." *International Conference on Machine Learning*. PMLR, 2019.
[19] https://towardsdatascience.com/on-object-detection-metrics-with-worked-example-216f173ed31e (pristupljeno 26.08.2021.)
[20] https://cocodataset.org/ (pristupljeno 26.08.2021.)
[21] http://shuoyang1213.me/WIDERFACE/ (pristupljeno 26.08.2021.)
[22] https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html (pristupljeno 26.08.2021.)
[23] https://susanqq.github.io/UTKFace/ (pristupljeno 26.08.2021.)
[24] https://www.image-net.org/ (pristupljeno 26.08.2021.)
[25] Chamoso, Pablo, et al. "Swarm agent-based architecture suitable for internet of things and smartcities." *Distributed Computing and Artificial Intelligence, 12th International Conference*. Springer, Cham, 2015.

Kratka biografija:

Dušan Bućan rođen je u Zrenjaninu 1997. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva odbranio je 2021.god.
kontakt: dusanbzr@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/15OI02Bosnjak

KONCEPTI I PRIMENA ROBOTSKE AUTOMATIZACIJE PROCESA

CONCEPTS AND APPLICATION OF ROBOTIC PROCESS AUTOMATION

Nataša Bošnjak, *Fakultet tehničkih nauka, Novi Sad*

Oblast – INŽENJERSTVO INFORMACIONIH SISTEMA

Kratak sadržaj – *U okviru rada opisani su osnovni koncepti robotske automatizacije procesa (RPA, engl. Robotic Process Automation) i karakteristike koje treba da poseduju poslovni procesi kako bi bili pogodni za robotsku automatizaciju. Objašnjene su faze razvoja rešenja upotrebom RPA, oblasti i prednosti primene robotske automatizacije u praksi i dat je pregled nekih primera uspešne primene RPA tehnologije u svetskoj industriji. Izvršeno je poređenje RPA sa metodologijom upravljanja poslovnim procesima. Analizirani su najpoznatiji alati za robotsku automatizaciju procesa i izabran je alat za realizaciju slučaja upotrebe kreiranja dnevnog izveštaja trgovanja akcije X na Beogradskoj berzi na kojem je prikazan postupak robotske automatizacije.*

Ključne reči: *Robotska automatizacija procesa, PRA, BPM, softverski robot*

Abstract – *This paper describes the basic concepts of Robotic Proces Automation (RPA) and the characteristics that business processes should have in order to be suitable for robotic automation. Development stages are explained by usage of RPA, as well as areas and advantages of RPA application and its benefits in industry. Paper also covers the comparation between RPA and Business Process Management (BPM). Major tools for RPA were analyzed and one of them was chosen for realization of use case for creating daily trading report for share X on the Belgrade Stock Exchange.*

Keywords: *Robotic process automation, RPA, BPM, software robot*

1. UVOD

Poslednjih decenija informacione tehnologije (IT, engl. *Information Technologies*) konstantno se razvijaju i sve su više prisutne u društvu, a samim tim utiču na kvalitet života i svakodnevno poslovanje. Mnoge kompanije širom sveta u raznim domenima poslovanja implementiraju različite softverske alate i tokove poslovnih procesa koji obuhvataju kombinaciju automatizovanog i manuelnog dela posla koji se izvršava od strane čoveka. Kako bi se unapredilo poslovanje, godinama unazad na poslovnom tržištu kompanije su u potrazi za softverskim tehnologijama koje bi omogućile povećanje produktivnosti, kvaliteta i satisfakcije zaposlenih, smanjenje operativnih troškova i oslobađanje zaposlenih od ponavljajućih aktivnosti.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bila dr Sonja Ristić, red. prof.

Četvrta industrijska revolucija dovela je do pojave tehnologija koje su usmerene na automatizaciju poslovnih procesa u čijem je fokusu značajno smanjenje učešća (pa čak i eliminacija, kada je to moguće) ljudskog rada i ovo se smatra glavnom promenom u odnosu na prethodne industrijske revolucije gde su tehnologije bile fokusirane na povećanje efikasnosti rada.

Robotska automatizacija procesa (RPA) je tehnologija usmerena na automatizaciju poslovnih procesa u kompanijama, i u kombinaciji sa veštačkom inteligencijom (AI, engl. *Artificial Intelligence*) i mašinskim učenjem (ML, engl. *Machine Learning*) omogućava kompanijama da ostvare svoje ciljeve u kontekstu smanjenja operativnih troškova i oslobađanja zaposlenih od ponavljajućih aktivnosti.

Osnovni cilj rada jeste unapređenje poslovnog procesa upotrebom RPA tehnologije i prikaz uštede vremena eliminacijom manuelnih ponavljajućih aktivnosti na realnom primeru. Kako bi se upoznali sa konceptima RPA tehnologije proučene su teorijske osnove robotske automatizacije procesa. Bilo je neophodno analizirati praktične aspekte primene u svetskoj industriji da bi se došlo do zaključka koji su procesi pogodni za primenu RPA tehnologije. Postoji više alata koji se koriste za RPA te je iz tog razloga izvršena analiza u cilju odabira najpogodnijeg alata u kom će biti robotizovan izabrani poslovni proces. Nakon detaljne analize i razumevanja koji su procesi pogodni za primenu RPA odabran je proces za robotizaciju, koji je obuhvatao *Kreiranje dnevnog izveštaja trgovanjem akcije X na Beogradskoj berzi* primenom odabranog *UiPath* alata. Postupak robotizacije odabranog procesa izvršen je saglasno fazama RPA životnog ciklusa.

Pored uvoda i zaključka rad sadrži još sedam poglavlja. U drugom poglavlju detaljno se objašnjava: pojam robotske automatizacije procesa, koji su procesi pogodni za primenu RPA tehnologije, faze implementacije rešenja i prednosti upotrebe RPA tehnologije. U trećem poglavlju izvršeno je poređenje robotske automatizacije procesa i upravljanja poslovnim procesima. Pregled i analiza aktuelnih RPA alata na tržištu objašnjeni su u četvrtom, dok je *UiPath* alat detaljnije opisan u petom poglavlju. Nakon što je izvršeno ispitivanje alata na tržištu, kroz šesto poglavlje prikazani su praktični primeri uvođenja RPA tehnologije u velikim kompanijama, i u sedmom poglavlju opisan je odabrani slučaj upotrebe nad kojim je primenjana RPA tehnologija.

2. ROBOTSKA AUTOMATIZACIJA PROCESA

Robotska automatizacija procesa je softverska tehnologija koja omogućava kreiranje, primenu i upravljanje

softverskim robotima koji oponašaju ljudske akcije u interakciji sa drugim softverima i informacionim sistemima [1, 2].

RPA tehnologija omogućava programerima da konfigurišu softverske robote tako da imitiraju korisnički rad na računarima i da, kao i ljudi razumeju, šta se nalazi na ekranskim formama, da izvrše određene komande, komuniciraju sa drugim sistemima, identifikuju i preuzmu podatke, ali na mnogo efikasniji i brži način, po pravilu bez nadgledanja od strane čoveka.

RPA tehnologija može značajno da utiče na digitalnu transformaciju, što dovodi do povećanja profitabilnosti, fleksibilnosti i skalabilnosti kompanija, a samim tim i povećanja zadovoljstva, angažovanja i produktivnosti zaposlenih kroz eliminaciju manuelnih ponavljajućih aktivnosti. Robotska automatizacija procesa koristi se za smanjivanje troškova poslovanja i povećanja profita, što proističe iz toga da jedan robot može da stvara značajne uštede.

Kako bi se postigle prednosti koje su prethodno objašnjene veoma je važno na pravi način odrediti procese pogodne za robotsku automatizaciju. Procesi koji su pogodni za RPA treba da budu rutinski, manuelni i ponavljajući, zasnovani na pravilima putem unapred definisane logike, pri čemu nije potrebna ljudska odluka prilikom izvršavanja aktivnosti.

Drugi važan kriterijum jeste da procesi budu digitalizovani, odnosno da koriste softverske aplikacije koje se nalaze na korisničkom računaru ili serveru [3]. Veoma je važno da procesi budu standardizovani, jasno i precizno definisani i da se izvršavaju svaki put na isti način. Procesi koji su podložni čestim promenama nisu pogodni za robotizaciju.

Prilikom odabira procesa pogodnih za RPA važno je fokusirati se na izbor onih koji su visoko frekventni, odnosno imaju veliki broj transakcija, poput procesa koji se izvode više puta tokom svakog dana, na dnevnom ili nedeljnom nivou.

Kako bi se uspešno kreirao softverski robot, neophodno je da se prođe kroz sledećih šest faza životnog ciklusa RPA.

1. Analiza – odabir odgovarajućih procesa za robotsku automatizaciju;
2. Dizajn rešenja – detaljno razumevanje procesa i dokumentovanje;
3. Razvoj – kreiranje skripte u odabranom RPA alatu od strane programera;
4. Testiranje – provera na testnom okruženju sa testnim podacima da li razvijeni robot radi u skladu sa definisanim zahtevima;
5. Prebacivanje na produkciono okruženje –okruženje na kom se nalazi poslednja verzija softvera koja je dostupna krajnjim korisnicima za upotrebu; i
6. Izvršavanje robota, korisnička podrška i održavanje – izvršavanje zadataka od strane robota uz konstantno nadgledanje performansi, održavanje i pružanje podrške krajnjim korisnicima za koje je izvršena robotska automatizacija.

3. UPRAVLJANJE POSLOVNIM PROCESIMA *VS*. ROBOTSKA AUTOMATIZACIJA PROCESA

Veoma često se pojmovi RPA i upravljanja poslovnim procesima (engl. *Bussines Process Management*, BPM) mešaju i dolazi do pogrešnog razumevanja njihove primene. Glavna razlika između ova dva pojma je što RPA drugim sistemima pristupa kroz prezentacioni sloj na isti način na koji to rade i ljudi, bez potreba za izmenama, unapređenjem i korigovanjem postojeće logike, kao i softverske i hardverske infrastrukture sistema. Sa druge strane BPM komunicira sa poslovnom logikom i slojem podataka koji je lociran iznad baze podataka. Važno je naglasiti da RPA ne zamenjuje BPM, već ga nadograđuje, odnosno nalazi se na višem nivou od BPM-a. Ovo potvrđuje činjenica da su kod BPM-a izlazi nove aplikacije i sistemi, dok kod RPA izlazi predstavljaju automatizacije postojećih sistema i aplikacija. Kombinacija RPA i BPM modela može da donese odgovarajuće prednosti kompanijama. BPM sistemi mogu da pozivaju RPA automatizovane aktivnosti koje vrše komunikaciju sa drugim informacionim sistemima kako bi koristili rezultate RPA procesa u podprocesnim koracima BPM toka. Robotska automatizacija procesa može da napravi odgovarajuće uštede u skupim BPM projektima postavljanjem RPA tehnologije na kritične tačke BPM toka procesa pri čemu će robotizacija obezbeditu veću tačnost i smanjenje verovatnoće pojave grešaka.

4. PREGLED ALATA ZA ROBOTSKU AUTOMATIZACIJU PROCESA

Postoje različiti alati za implementaciju robotske automatizacije. U okviru rada opisana su tri alata kompanija koje se nalaze u Gartnerovom magičnom kvadrantu na poziciji lidera. Pregled osnovnih karakteristika dat je u tabeli 1. Pored naziva kompanije, godine osnivanja, sedišta kompanije u tabeli su prikazane i tehnologije na kojima su zasnovani predstavljeni alati. U tabeli su prikazane i osnovne komponente alata, pri čemu se svi alati sastoje uglavnom od komponente za modelovanje procesa, orkestratora čija je uloga nadgledanje, upravljanje i izvršavanje softverskih robota i naprednih alata koji zavise od samog proizvođa. *UiPath* i *Automation Anywhere* imaju mogućnost snimanja koraka procesa i pretvaranja u pozadinsku skriptu, kao i izvršavanja *front offfice* i *back office* procesa dok alat kompanije *Blue Prism* ne poseduje mogućnost snimanja, i namenjen je za izvršavanje samo *back office* procesa koji nemaju interakcije sa klijentima. Autor rada imao je najviše iskustva u radu sa alatom kompanije *UiPath* koji je odabran za realizaciju slučaja korišćenja.

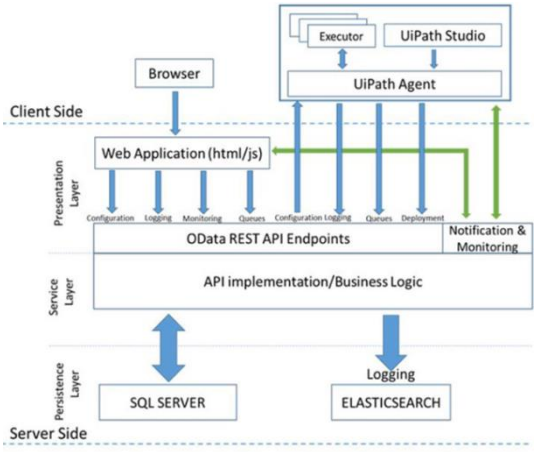
5. ANALIZA UIPATH ALATA

UiPath arhitektura podeljena je na klijentsku i serversku stranu, pri čemu serverska strana sadrži tri sloja: prezentacioni sloj, servisni sloj i sloj koji obezbeđuje komunikaciju sa bazama podataka. Na slici 1 prikazana je *UiPath* arhitektura. Tri osnovne komponente *UiPath* platforme su *UiPath Studio*, *UiPath Robot* i *UiPath Orchestrator*. Studio predstavlja napredni softver za automatizaciju koji omogućava biznis korisnicima i naprednim programerima da konfigurišu softverske robote upotrebom gotovih komponenti koje se prevlače. Roboti su izvršioci procesa koji su kreirani u studiu i koji mogu da se izvršavaju sa računara ili orkestratora. *UiPath* podržava rad sa nadgledanim robotima koji ne mogu da se

izvršavaju ukoliko je korisnički ekran zaključan i sa nenadgledanim robotima koji se koriste za izvršavanje kompleksnijih aktivnosti i ne zahtevaju nadgledanje od strane čoveka. Orkestrator predstavlja centralni deo za upravljanje automatizacijama, odnosno robotima.

Tabela 1 – *Osnovne karakteristike RPA alata*

Kompanija	***UiPath***	***Automation Anywhere***	***Blue Prism***
Godina osnivanja kompanije	2005.	2003.	2001.
Sedište kompanije	Njujork	San Hoze	Varington
Osnovne komponente	Studio, robot i orkestrator	Kreator procesa, izvršilac procesa i kontrolna soba	Procesni dijagram, procesni studio, objektni studio i komponenta z a modelovanje aplikacija
Tehnologija	*Sharepoint, Kibana* i *Elasticsearc h*	*Microsoft* tehnologije	*C#*
Mogućnost snimanja	Da	Da	Ne
Softverski roboti	Izvršavanje i *front office* i *back office* procesa	Izvršavanje i *front office* i *back office* procesa	Izvršavanje samo *back office* procesa



Slika 1. *UiPath arhitektura* [4]

6. PRIMENA ROBOTSKE AUTOMATIZACIJE PROCESA U PRAKSI

Na osnovu prikupljene literature opisani su neki od uspešnih primera uvođenja RPA tehnologije u kompanijama širom sveta. *Telefónica O2* drugi najveći provajder telekomunikacionih usluga na području Velike Britanije sa 15 robotizovanih poslovnih procesa brojala je oko 400,000 do 500,000 transakcija na mesečnom nivou koje su se izvršavale od strane 160 robota. Uložen novac u investicije uvođenja RPA povraćen je nakon 12 meseci, dok će povrat investicija za 3 godine iznositi od 600 do 800%. Drugi uspešan primer je kompanija *Xchanging*, provajder tehničkih i poslovnih usluga koja je po podacima iz 2015-te godine brojala 15 robotizovanih poslovih procesa, odnosno 120,000 transakcija na mesečnom nivou koje je izvršavalo 27 robota. Odlične rezultate uvođenjem RPA tehnologije postigla je i kompanija *Skanska*, jedna od vodećih kompanija u građevinskom sektoru. U periodu kraćem od tri godine, *Skanska* je uspela da robotizuje 35 poslovnih procesa, što je dovelo do uštede od 10,000 radnih sati. Proces obrade zarada na koji su zaposleni trošili celu nedelju, nakon robotizacije se izvršava za 5 minuta. Još jedan primer uspešne implementacije robotske automatizacije je *YapiKredi*, treća najveća privatna banka u Turskoj. Za kratak vremenski period banka je uspela da robotizuje 137 procesa koji se izvršavaju od strane 20 nenadgledanih robota bez nadzora od strane zaposlenih. Fascinantnih 4 miliona transakcija u 2020-toj godini i 2,2 miliona transakcija u prvom kvartalu 2021. godine izvršeno je od strane robota. Robotizacijom poslovnih procesa u svim kompanijama povećan je kvalitet usluga i smanjen je broj grešaka, odnosno povećana je tačnost rezultata. Navedene kompanije posluju u različitim industrijskim oblastima, što dovodi do zaključka da RPA omogućava značajne rezultate bez obzira na delatnost poslovanja.

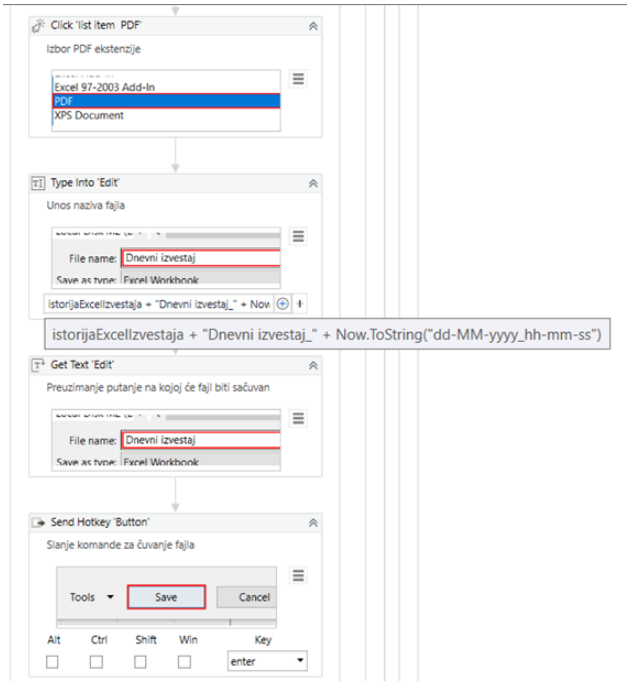
7. IMPLEMENTACIJA ROBOTIZOVANE AUTOMATIZACIJE ODABRANOG PROCESA

U ovom poglavlju prikazan je primer robotizacije procesa kreiranja izveštaja o trgovanju određene akcije X na Beogradskoj berzi. Aktivnosti koje su izvršavane manuelno obuhvatale su prepisivanje cene, promene, obima, prometa, cene otvaranja, najviše dnevne cene, najniže dnevne cene i količinski ponderisane prosečne cene (engl. *Volume-Weighted Average Price*, VWAP) sa sajta Beogradske berze, zatim preuzimanje kurseva dolara i evra sa sajta Narodne banke Srbije i ukupnog broja akcija sa odgovarajućeg izvora podataka. Sa sajta Beogradske berze, potrebno je preuzeti i grafikon o trgovanju akcijama tokom poslednjih 30 dana. Nakon preuzimanja podataka, vrši se izračunavanje prometa u dolarima i evrima, popunjavanje i kreiranje izveštaja i slanje na određene *email* adrese. Izveštaj se kreira svakog radnog dana, odnosno njegova frekvencija kreiranja je u proseku 22 puta mesečno. Za kreiranje izveštaja u manuelnom režimu rada potrebno je oko 30 minuta.

Kompletno rešenje razvijeno je u *UiPath Studiu* dok je proces namenjen za izvršavanje na nadgledanom robotu. Proces je linearan, nema odlučivanja, aktivnosti se izvršavaju jedna za drugom i iz tog razloga kompletna logika implementirana je u okviru sekvence. Tok procesa organizovan je u pet delova, preuzimanje podataka sa sajta Beogradske berze, preuzimanje podataka sa sajta Narodne banke Srbije, preuzimanje broja ukupnih akcija sa odgovarajućeg izvora podataka, upisivanje preuzetih podataka i deo kreiranja izveštaja i interakcije sa korisnicima.

Za kreiranje toka procesa najčešće korišćene komponente su *Open Browser* koja se koristi za otvaranje sajta u definisanom internet pretraživaču, *Maximize Window* za povećanje prozora pretraživača na maksimum kako bi robot uspešno izvršio aktivnosti preuzimanja podataka. *Type Into* komponenta korišćena je za upis teksta u odgovarajuća polja, komponenta *Get Text* korišćena je za

preuzimanje teksta sa odgovarajućih lokacija. Za rad sa *excel* fajlom korišćena je komponenta *Write Cell* za upis podataka u fajl, *Read Cell* za čitanje vrednosti iz ćelije, a *Read Range* za čitanje opsega vrednosti. Za potrebe slanja skraćenih komandi sa tastature korišćena je komponenta *Send Hotkey*. *Click* komponenta koristi se za obeležavanje komponenti korisničkog interfejsa gde robot treba da klikne. Za slanje *email*-a korišćena je *Send SMTP Mail Message* komponenta. U samim komponentama pisan je kôd koji je bio potreban za izvršavanje logike toka procesa. Na slici 2 prikazan je deo toka procesa u kom su povezane neke od prethodno objašnjenih komponenti.



Slika 2. *Prikaz dela toka procesa u UiPath alatu*

Vreme koje je bilo potrebno za kreiranje izveštaja, upotrebom robotske automatizacije redukovano je sa 30 minuta u slučaju kada čovek izvršava proces, na vreme koje u proseku iznosi 1 minut. Robotizacijom procesa kreiranja izveštaja vreme je skraćeno za 10,5 sati na mesečnom nivou. Na slici 3 prikazan je PDF izveštaja koji je kreiran od strane robota.

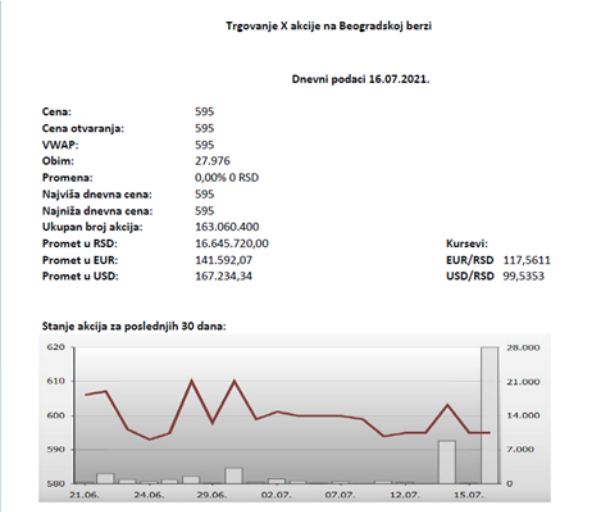
9. ZAKLJUČAK

Rezultati primene RPA tehnologije na tržištu doveli su do povećanja efikasnosti poslovanja i samim tim privukli su pažnju velikih kompanija širom sveta. RPA tehnologija podstiče kompanije da standardizuju poslovne procese što dovodi do smanjenja verovatnoće pojave grešaka, povećanja kvaliteta podataka i pouzadnijih analiza. Iz ličnog iskustva autora rada, najvažnija je dobro razumeti proces koji je potrebno robotizovati i izvršiti njegovu detaljnu analizu.

Podaci i informacije koje su sklone promenama, a koji se koriste od strane robota bolje je čuvati u fajlovima poput *excel*-a radi lakše korekcije i održavanja robota. U toku analize zahteva, potrebno je proveriti da li postoje planovi za izmenu toka procesa, aplikacija i izvora podataka koji se koriste u procesu u nekom realnom vremenskom intervalu koji obuhvata raspon od 6 do 9 meseci. Ukoliko postoji mogućnost za izmenama, najbolje je odložiti realizaciju zahteva dok se proces ne izmeni i ne stabilizuje. Nije uvek moguće uočiti verovatnoću promena izvora podataka. Kod eksternih sajtova i aplikacija koje nisu u domenu održavanja kompanije koja želi da automatizuje svoje procese nije moguće pouzdano predvideti da li će u određenom vremenskom periodu postojati izmene i kojeg će obima one biti. Odabrani proces *Kreiranje dnevnog izveštaja trgovanjem akcije X na Beogradskoj berzi* realizovan je u *UiPath* alatu.

Vreme kreiranja izveštaja na mesečnom nivou smanjeno je za 10,5 sati. Ukoliko bi se javila potreba za dodatnim podacima koje je potrebno prikazati na dnevnom izveštaju moguće je izvršiti korekcije na veoma brz i jednostavan način.

Pravci daljeg istraživanja mogu da obuhvate temu jasnog definisanja metodološkog postupka određivanja pogodnih procesa za robotizaciju, kao i primenu veštačke inteligencije i drugih naprednih mogućnosti nad robotskom automatizacijom kompleksnijeg poslovnog procesa.



Slika 3. *PDF izveštaj izgenerisan od strane robota*

10. LITERATURA

[1] J. Ribeiro, R. Lima, T. Eckhardt, S. Paiva, "*Robotic Process Automation and Artificial Intelligence in Industry 4.0 – A Literature review*", Instituto Politécnico de Viana do Castelo, Portugal, 2020.

[2] https://www.uipath.com/rpa/robotic-process-automation (pristupljeno u maju 2021.)

[3] https://hohmannchris.wordpress.com/2020/09/27/-what-processes-are-suitable-for-rpa-robotic-process-automation/ (pristupljeno u junu 2021.)

[4] https://forum.uipath.com/t/the-technical-architecture-of-uipath/11599/4 (pristupljeno u junu 2021.)

Kratka biografija:

Nataša Bošnjak rođena je u Somboru 1996. godine. Fakultet tehničkih nauka upisala je 2015. godine. Diplomski rad iz oblasti Informacionih tehnologija odbranila je 2019. godine.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004:007
DOI: https://doi.org/10.24867/15OI03Nedeljkovic

INFORMACIONI SISTEM ZA PODRŠKU IZRADI PLANA ISHRANE

INFORMATION SYSTEM FOR SUPPORT OF CREATING NUTRITION PLAN

Milica Nedeljković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – INŽENJERSTVO INFORMACIONIH SISTEMA

Kratak sadržaj– *U ovom radu je opisan predlog softverskog rešenja za podršku kreiranja plana ishrane u trajanju od osam nedelja. Osim pružanja usluge krajnjim korisnicima – pacijentima, nutricionistima je olakšana izrada plana, kao i korišćenje baze podataka sa već postojećim receptima. Predstavljene su tehnologije korišćene prilikom izrade aplikacionog rešenja, kao i funkcionalnosti koje su na raspolaganju. Predložene su mogućnosti unapređenja aplikacije, sa ciljem stabilizacije sistema, interakcije sa korisnicima i mogućnosti profita.*

Ključne reči: *.NET Core, ASP.NET Core, EF Core, SQL, REST, Angular, autentifikacija, autorizacija, plan ishrane*

Abstract – *The paper describes a proposal for a software solution to support the creation of a diet plan for a period of eight weeks. In addition to providing services to end users - patients, nutritionists it is made easier to create a plan, as well as use a database with existing recipes. The technologies used in the development of the application solution are presented, as well as the functionalities that are available. Besides that, the possibilities to improve the application are proposed, with the aim of stabilizing the system, interaction with users and profit opportunities.*

Keywords: *NET Core, ASP.NET Core, EF Core, SQL, REST, Angular, Autentification, Autorization, meal plan*

1. UVOD

Sve veći broj ljudi okreće se zdravom životu, koji obuhvata zdravu ishranu i fizičku aktivnost. Zdrava ishrana oduvek je predstavljala veliki izazov za svakog od nas, najčešće zbog loše organizacije i nedostatka znanja. Povrh svega ona je i dalje sinonim za velika odricanja i teške dijete, što mnoge demotiviše da se okrenu ovakvom načinu života.

Osnovne funkcionalnosti koje aplikacija pruža jeste kreiranje plana ishrane u trajanju od osam nedelja, praktičan prikaz recepata i mogućnost praćenja sopstvenog napretka. Realizovana je tako da bude jednostavna za upotrebu i prilagodljiva bilo kome. Cilj aplikacije jeste da nakon osam nedelja korisnik pored postignutih rezultata stekne zdrave navike i nastavi sa ovakvim načinom života.

Specifičnost ove aplikacije ogleda se i u mogućnosti korisnika da ukoliko želi da izdvoji novac može da dobije plan ishrane specijalno osmišljen za njega od strane nutricioniste kojeg je sam izabrao. Svi nutricionisti koji se pretplate na aplikaciju mogu biti izabrani od strane korisnika, na taj način reklamiraju sebe i stiču nove pacijente.

Aplikacija pruža podršku nutricionistima za vođenje evidencije pacijenata i formiranje plana ishrane, pri čemu mogu koristiti postojeće recepte iz baze ili dodavati nove, što ujedno čini planove ishrane kreirane od strane algoritma raznovrsnijim.

2. KORIŠĆENE TEHNOLOGIJE

Aplikacija je bazirana na arhitekturalnom obrascu klijent-server, razvojno okruženje je Visual Studio 2019 Community Edition, dok se komunikacija sa bazom odvija pomoću SQL Server-a 2019. U nastavku je dat kratak opis korišćenih tehnologija.

2.1 .NET

Stvoren od strane *Microsoft*-a, predstavlja okruženje za razvoj softvera sa velikim brojem biblioteka i komponenti koje omogućavaju da se kod izvršava. Programi se izvršavaju kroz softversko okruženje CLR (*Common Language Runtime)*, virtualnu mašinu koja sadrži: upravljanje memorijom, izuzecima [1]. Omogućeno je korišćenje dvadeset i pet programskih jezika od kojih su najpolularniji *C#*, *C++* i *VisualBasic*.

2.2 .NET Core

.NET Core je nova verzija .NET Framevork-a, koja je besplatna, otvorenog koda (engl. *open-source*), opšta namena razvojne platforme koju održava *Microsoft*. Podržava različite tipove aplikacija - na .NET Core platformi mogu se razviti i pokretati razne vrste aplikacija kao što su mobilni uređaji, računari, web, oblak, IoT (*Internet of Things*), mašinsko učenje, mikroservisi, igre itd. Radi na operativnim sistemima *Windows*, *macOS* i *Linux*.

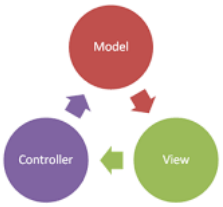
2.3 ASP.NET Core

ASP.NET Core je višeplatformski okvir, otvorenog koda, visokih performansi za izgradnju modernih aplikacija, povezanih na internet, omogućenih na *Cloud*-u. Izvršavaju se na .NET Core-u pomoću razvojnih alata mogu se razvijati na Windows-u, MacOS-u i Linux-u.[2]

2.4 ASP.NET Core MVC

Predstavlja okruženje koje se koristi za razvoj web aplikacija, na osnovu MVC (*Model-View-Controller)* dizajn paterna.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Srđan Sladojević, vanr. prof.



Slika 2.1 – MVC dizajn obrazac

MVC dizajn obrazac je arhitekturalni patern koji aplikaciju deli na tri celine:

- *Model* predstavlja strukturu i trenutno stanje podataka aplikacije, kao i svu poslovnu logiku koja se odnosi na promenu stanja podataka.
- *Controller* je komponenta koja upravlja interakcijom korisnika. Predstavlja početnu tačku pristupa i odgovoran je za odabir tipova modela za rad koji se prikazuju na korisničkom interfejsu.
- *View* reprezentuje informacije u određenom formatu.

2.5 Entity Framework Core

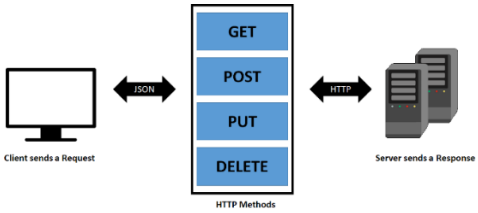
Predstavlja višeplatformsku, open-source verziju EF tehnologije pristupa podacima, dizjaniranu da bude jednostavna i proširiva.
EF Core je ORM (*Object-Relational Mapping*). Objektno-relaciono mapiranje je tehnika koja omogućava programerima da rade sa podacima na objektno orijentisan način obavljajući posao potreban za mapiranje između objekata definisanih u programskom jeziku aplikacije i podataka uskladištenih u relacionim izvorima podataka

2.6 SQL Server Management Studio (SSMS)

SSMS je integrisano okruženje koje omogućava korisnički interfejs za konekciju i rad sa MS SQL serverom. Služi za upravljanje bilo kojom SQL infrastrukturom, od SQL servera do Azure SQL baze podataka [3].

2.7 REST

REST (Representational State Transfer) predstavlja arhitekturalni stil za dizajniranje web aplikacija.
REST arhitektura se zasniva na prenosu podataka između klijenta i servera putem HTTP/HTTPS protokola i podaci se prenose najčešće u JSON (*JavaSript Object Notation*) formatu, mada su podržani XML i YAML formati. Zbog svoje jednostavnosti i fleksibilnosti, RESTful servis se izdvojio kao vodeći.[4]



Slika 2.2 – REST arhitekturalni stil

2.8 Angular

Angular je radno okruženje programskog jezika *JavaScript*, i u novijim verzijama *TypeScript*. Razvile su ga kompanije *Google* i zajednice pojedinaca i drugih kompanija. Pruža razvoj dobro struktuiranih jednostraničnih veb aplikacija (eng. *Single Page Application*).
Jednostranična aplikacija je veb aplikacija ili veb sajt čiji se ceo sadržaj nalazi na jednoj strani, svi neophodni resursi za izvršavanje se učitavaju pri učitavanju aplikacije, kasnije se kroz interakciju sa korisnikom učitavaju ostali resursi po potrebi. Poslovna logika ovakvih aplikacija sada se ne nalazi samo na serverskoj nego i na klijentskoj strani.
Okruženje dolazi sa komandnim alatom *Angular-CLI* koji olakšava inicijalizaciju i kreiranje pravilne strukture aplikacije i određenim komandama ubrzava sam razvoj aplikacije.

2.9 Autentifikacija i autorizacija

Autentifikacija i autorizacija su dva usko povezana termina koji se koriste radi postizanja sigurnosti u distribuiranim aplikacijama. Autentifikacija predhodi autorizaciji i odnosi se na verifikaciju identiteta korisnika. Autorizacija daje ovlašćenja autentifikovanim korisnicima da pristupaju određenim resursima.

2.9.1 Autentifikacija

Za kreiranje bilo kog naloga neophodno je korisničko ime i lozinka kao najjednostavniji vid verifikacije, a svako naredno prijavljivanje na taj nalog uključuje tačan unos registrovanih kredencijala.
Kod autentifikacije postoji nekoliko nivoa zaštite koji se mogu primeniti u zavisnosti koliko su poverljivi podaci koje korisnik ima na svom nalogu: jednofaktorna, dvofaktorna i višefaktorna autentifikacija [5].

2.9.2 Autorizacija

Autorizacija je proces koji sledi nakon autentifikacije i pomoću nje se korisnicima dodeljuju prava pristupa na mreži, bazi podataka, servisima i ostalim resursima. Jako je važno napomenuti da su autentifikacija i autorizacija neophodni zajedno kako bi sistem bio osiguran. Korisnik se može identifikovati nakon autentifikacije, međutim da bi mogao da koristi neki informacioni sistem, neophodno mu je dati ovlašćenja.
Svako poslovno okruženje, bilo malo ili veliko poseduje osetljive podatke koji ne treba da budu dostupni svima. Kako bi se očuvala privatnost, korisnici se razvrstavaju po grupama koje su definisane polisama i na taj način smo sigurni da će podaci ostati zaštićeni.

2.9.3 JSON Web Token

JWT (*JSON Web Token*) je otvoreni standard *RFC 7519* koji definiše način za bezbedan i siguran prenos podataka u obliku JSON objekta. [6] RFC (*Request For Comment*) je formalni dokument izrađen od strane IETF (*Internet Engineering Task Force*) koji se koristi za opisivanje specifikacije određene tehnologije [7].

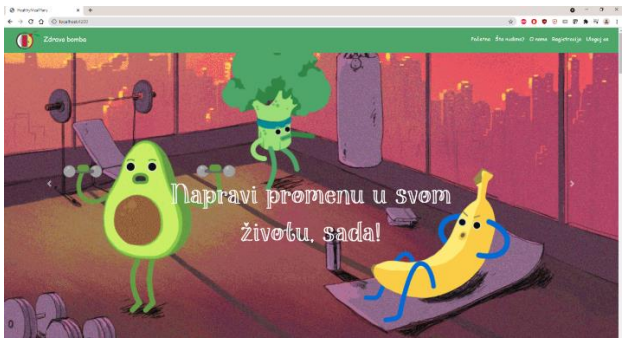
3. OPIS FUNKCIONALNOSTI SISTEMA

Aplikacija se može posmatrati iz tri ugla: korisnika (pacijenta), nutricioniste i administratora.

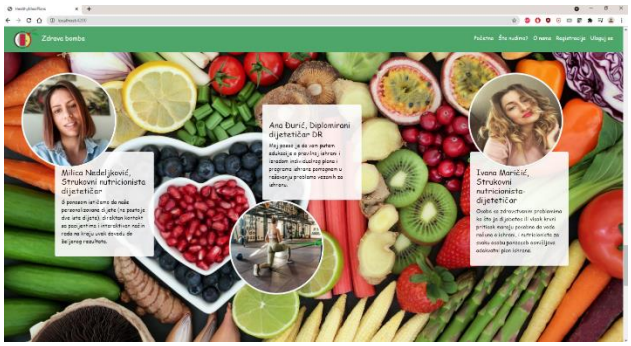
3.1 Pacijent

Početna stranica prikazuje kroz slike i kratke poruke šta korsinik može da očekuje od aplikacije, koji paket i

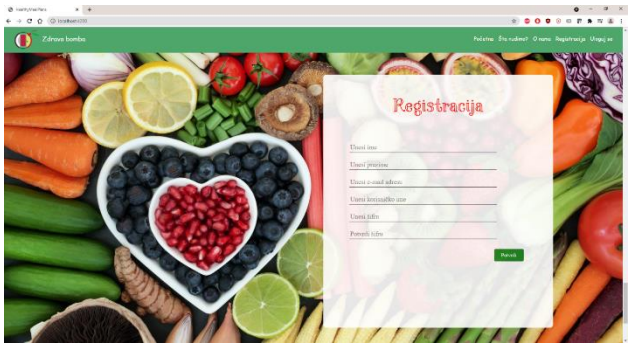
nutricionistu može da izabere (Slika 3.1.2), primer jelovnika, kao i mogućnost registracije.



Slika 3.1.1 – Početna stranica – prvi deo



Slika 3.1.2 – Početna stranica – prikaz nutricionista



Slika 3.1.3 – Početna stranica – registracija

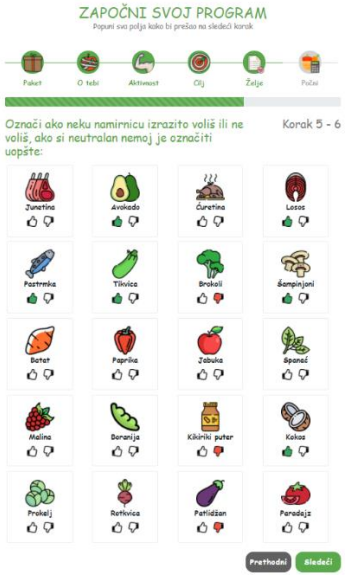
Registracija započinje na početnoj stranici (Slika 3.1.3), tako što korisnik unosi svoje osnovne podatke navodeći korisničko ime i šifru, tada se njegov nalog kreira i otvara se sledeća stranica koja čini drugi deo registracije i sastoji se od nekoliko koraka kroz koje korisnik treba da prođe kako bi se uspešno registrovao.

U prvom koraku korisnik bira da li želi biti *prime* (premium) ili regularan korisnik. Regularan korisnik dobija plan ishrane koji generiše algoritam na osnovu unetih informacija, dok prime korisnik ima mogućnost izbora nutricioniste, zatim mu izabrani nutricionista kreira plan ishrane. Prime usluga se naplaćuje 3500 dinara.

Drugi korak zahteva od korisnika da unese osnovne informacije o sebi kao što su: pol, visina, težina i datum rođenja.

U trećem koraku utvrđujemo koliki je stepen fizičke aktivnosti korisnika. Ponuđene su mu sledeće opcije:

- Minimalna aktivnost, bez treninga.
- Lagano vežbanje, trening 1-3 dana u nedelji.
- Umereno vežbanje, trening 3-5 dana u nedelji.
- Intenzivno vežbanje, trening 6-7 dana nedeljno.
- Veoma intenzivno vežbanje, trening više od jednom dnevno.



Slika 3.1.4 – Selekcija namirnica na osnovu korisnikovih želja

Korisnik na osnovu svog trenutnog telesnog stanja bira koji je njegov krajnji cilj, odnosno šta je ono što želi da postigne. Recimo njegov cilj može biti da samo održava trenutnu telesnu težinu ili da smanji procenat masnog tkiva i izgradi mišićnu masu.

Kako bi jelovnik bio prilagođen korisniku, omogućeno mu je da od ponuđenih namirnica označi ukoliko ih želi ili ne želi u svom jelovniku (Slika 3.1.4).

Prime korisnik ima dva dodatna koraka u odnosu na regularnog. Jedan korak jeste izbor nurticioniste, a drugi zahteva unos informacija neophodnih za plaćanje usluge.

Nakon završene registracije korisnik dobija konfimacioni mejl, koji sadrži link i neophodno je da klikne na njega, kako bi bio identifikovan. Klikom na link otvara mu se *login* stranica.

Kada se korisnik uloguje biće mu prikazana stranica koja sadrži određena pravila kojih korisnik treba da se pridržava tokom trajanja plana, kao i edukativne informacije koje će mu pomoći da i u budućnosti vodi zdrav život.

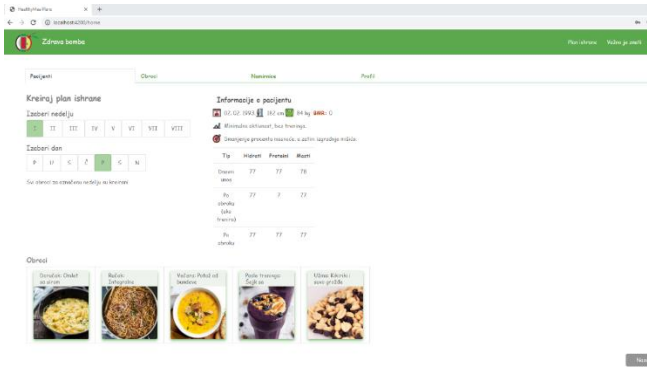
Glavna stranica aplikacije prikazuje plan ishrane za tekuću nedelju (Slika 3.1.5.). Korisnik uvek ima uvid u tekuću i prethodne nedelje, dok će mu sledeća nedelja biti dostupna dva dana pre njenog početka. Klikom na obrok, otvara se novi prozor na kom je prikazan recept obroka.

Slika 3.1.5 – Prikaz plana ishrane

Korisnik ima uvid u svoj profil na kom je prikazana raspodela nutrienata koje treba da unese dnevno, kao i praćenje svog napretka na osnovu predhodno uneih mera.

3.2 Nutricionista

Nakon logovanja nutricionisti se prikazuje početna strana aplikacije iz ugla nutricioniste Nutricionista ima uvid u tabelu sa pacijentima, mogućnost kreiranja plana ishrane za određenog pacijenta (Slika 3.2.1), dodavanje novih namirnica i obroka u bazu podataka.



Slika 3.2.1 – Kreiranje plana ishrane

Nutricionista može da pristupi svom profilu na kom su prikazane njegove lične informacije i tekst kojim će se predstaviti pacijentima, kao i mogućnost njihove izmene.

3.3 Administrator

Prvi jezičak aplikacije administratora prikazuje sve pacijente odabranog nutricioniste, bez mogućnosti kreiranja plana ishrane, dok su jezičak "Obroci" i "Namirnice" potpuno isti kao u aplikaciji nutricioniste. Osnovna funkcionalnost administratora jeste da dodaje novog nutricionistu. U jezičku "Nutricionisti" administrator ima uvid u postojeće nutricioniste, njihove profile i mogućnost pretrage. Dodavanje se izvršava tako što administrator unese mejl nutricioniste, na koji će biti poslat mejl sa linkom za njegovu registraciju.

4. ZAKLJUČAK

Pre postavljanja aplikacije u rad potrebno je izvršiti testiranje i dodatne kosultacije sa nutricionistima, kako bi algoritam koji kreira plan ishrane bio što sigurniji i precizniji. Posebno pažnju prilikom testiranja treba obratiti na brzinu izvršavanja algoritma ukoliko baza sadrži veliki broj recepata, zatim shodno ishodu testiranja refaktorisati trenutni algoritam.

Aplikacija je izrazito pogodna za različite vrste unapređenja. Forum bi omogućio korisnicima da započinju različite teme i dobijaju odgovore od strane stručnih lica, u ovom slučaju nutricionista. Pored nurtricionista, profesionalni treneri danas često nude paket u kojima kreiraju planove ishrane zajedno sa programnom treninga za određeni period, kako bi njihovi klijenti imali kompletan program i postigli što bolje rezultate.

Nadogradnja aplikacije stoga bi mogla ići u smeru pružanja podrške trenerima za kreiranje i prikazivanje plana treninga. Treneri bi u tom slučaju takođe bili pretplaćeni na aplikaciju, a njihova dobrobit bi bila reklama i sticanje novih klijenata.

Kako bi treneri i nutricionisti sa svojim klijentima ostvarili bolju saradnju u vidu bolje informisanosti o samom napretku i problemima s kojima se klijenti susreću tokom trajanja programa, poželjno bi bilo implementirati određeni vid "chata", odnosno servisa za razmenjivnanje poruka.

Takođe, trenutna funkcionalnost aplikacije koja formira nedeljni spisak namirnica za kupovinu, mogla bi biti povezana sa *online* marketima i omogućiti korisnicima lakšu nabavku sastojaka. Pored unapređenja, aplikacija je pogodna za sve vrste reklama i sponzorisanja različitih prehrambenih i sportskih proizvoda.

5. LITERATURA

[1] https://en.wikipedia.org/wiki/.NET_Framework, datum poslednjeg pristupa 06.09.2021.

[2] https://www.tutorialsteacher.com/core/dotnet-core, datum poslednjeg pristupa 06.09.2021.

[3] https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15, datum poslednjeg pristupa 06.09.2021.

[4] https://www.redhat.com/en/topics/api/what-is-a-rest-api, datum poslednjeg pristupa 06.09.2021.

[5] https://medium.datadriveninvestor.com/authentication-vs-authorization-716fea914d55, datum poslednjeg pristupa 06.09.2021.

[6] https://jwt.io/introduction, datum poslednjeg pristupa 28.06.2021.

[7] Request for Comment (RFC) https://www.techopedia.com/definition/27929/request-for-comments-rfc, datum poslednjeg pristupa 06.09.2021.

Kratka biografija:

Milica Nedeljković rođena je u Šapcu 1991. godine. Osnovne studije na Fakultetu tehničkih nauka iz oblasti Inženjerstvo informacionih sistema odbranila je 2017. godine.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004:007
DOI: https://doi.org/10.24867/15OI04Babic

SISTEM ZA AUTOMATIZACIJU PROCESA RASPOREĐIVANJA ZAPOSLENIH

SYSTEM FOR AUTOMATION OF EMPLOYEE SCHEDULE PROCESS

Zorica Babić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – INŽENJERSTVO INFORMACIONIH SISTEMA

Kratak sadržaj– *Dobar raspored i organizacija kompanije osigurava da se poslovi uvek završavaju na vreme i da poslodavac sa jedne i zaposleni sa druge strane budu zadovoljni. Kako bi se rukovodiocima olakšao posao i kako ne bi dolazilo do nekih propusta, osmišljeni su sistemi koji na osnovu određenih kriterijuma raspoređuju zaposlene po poslovima (zadacima) unutar kompanije. U ovom radu opisan je jedan takav sistem, kao i tehnologije koje su korišćene za izradu tog sistema.*

Ključne reči: *.NET, Angular, Upravljanje ljudskim resursima, Upravljanje zaposlenima, Raspored zaposlenih, web aplikacija*

Abstract – *Good schedule and organization of a company ensure that jobs will be done on time and that both the employer and employees are satisfied. To make managers' job easier and to avoid omissions, there are systems that delegate jobs (tasks) among employees according to various criteria. In this paper, that kind of systems will be described, as well as the technologies used for the development of those systems.*

Keywords: *.NET, Angular, Human Resource Management, Employee Management, Employee Rostering, Web application*

1. UVOD

Ljudski resursi predstavljaju najvažniji element organizacione strukture kompanije, pa se osnova uspešnosti ogleda u njenim zaposlenima. Njihove veštine i dostupnost doprinose ostvarivanju ciljeva kompanije. Kao i nad svim ostalim delovima organizacije, tako je potrebno upravljati i ljudskim resursima kako bi se uspešnije ostvarila strategija i realizovali planovi organizacije. Dodatno, dobra organizacija ljudskih resursa može biti korisna kada su u pitanju izazovi i zahtevi sa kojima se kompanija često suočava.

Upravljanje ljudskim resursima (engl. *Human Resource Management*) predstavlja skup aktivnosti koji su usmereni prema zaposlenima, a koji se baziraju na popunjavanju radnih mesta ljudima čiji se potencijal može maksimalno iskoristiti odnosno ljudima koji su kompatibilni sa opisom posla koji treba da obavaljaju [1]. Težina upravljanja ljudskim resursima kompanije zavisi od broja zaposlenih, obima njihovog posla, kao i frekvencije menjanja vrste posla unutar iste kompanije.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Srđan Sladojević.

Da bi se ovakvo upravljanje olakšalo, kreirani su mnogi sistemi koji omogućavaju automatizaciju upravljanja ljudskim resursima. Osnovne funkcije ovih sistema su ažuriranje podataka o zaposlenima, održavanje organizacione šeme preduzeća i integracija sa drugim relevantnim sistemima [2].

Na osnovu podataka o zaposlenom, koji postoje zahvaljujući upravljanju ljudskim resursima, mogu se doneti odluke o tome koji zaposleni je odgovarajući za određenu vrstu posla. Za donošenje ovakvih odluka zaduženo je još jedno upravljanje, a to je upravljanje zaposlenima (engl. *Employee Management*).

Upravljanje zaposlenima je holistički proces koji pokriva sve što je povezano sa ljudskim resursima kako bi zaposleni što bolje obavljali svoj posao i time ostvarili poslovne ciljeve. Postoje tri ključne oblasti koje pokriva upravljanje zaposlenima, a to su pronalaženje zaposlenih i zapošljavanje, angažman i zadržavanje i upravljanje učinkom [3].

Fokus se stavlja na angažman i zadržavanje zaposlenih što podrazumeva angažovanje zaposlenih na one poslove čiji opis odgovara karakteristikama i veštinama koje zaposleni poseduje, uz obezbeđenje da ne bude opterećen prevelikim obimom posla.

Zbog mogućeg velikog obima posla i velikog broja zaposlenih u kompanijama, potrebno je koristiti sisteme koji omogućavaju lakše raspoređivanje zaposlenih.

To su takozvani pametni sistemi koji na osnovu različitih vrsta konfiguracija unutar njega obezbeđuju to da odgovarajući zaposleni bude na najbolji mogući način raspoređen po poslovima u okviru nekog vremenskog opsega.

2. KORIŠĆENE TEHNOLOGIJE I ALATI

2.1. ASP.NET Core

ASP.NET Core predstavlja *framework* otvorenog koda, visokih performansi koji služi za izgradnju modernih *web*, IoT (*Internet of Things*) i mobilnih aplikacija. Ovaj *framework* predstavlja verziju ASP.NET-a koja radi na *macOS*-u, *Linux*-u i *Windows*-u [4]. Dizajniran je po modulima sa minimalnim troškovima, a dodatne funkcije se mogu instalirati kao *NuGet* paketi. Takav način dizajniranja obezbeđuje visoke performanse, zahteva se manje memorije i lako održavanje.

2.2. Entity Framework

Entity Framework komponenta u okviru .NET platforme obezbeđuje ORM (*Object Relational Mapping*). On

pojednostavljuje pristup podacima tako što omogućava pisanje koda za kreiranje, čitanje, ažuriranje i brisanje podataka. *Entity Framework* omogućava programerima da se bave podacima u formi objekata i odgovarajućih karakteristika, umesto da direktno barataju tabelama i kolonama baze podataka. Prilikom izvršavanja CRUD (*Create*, *Read*, *Update*, *Delete*) operacija nema potrebe za direktnom interakcijom sa bazom podataka. Ovo je omogućeno time što *Entiy Framework* nudi tri pristupa za modeliranje entiteta sistema, a to su *Model First*, *Database First* i *Code First* pristup.
Razlog velike popularnosti *Entity Framework*-a leži u njegovoj sposobnost da veliki deo koda generiše automatski i na taj način programerima štedi vreme [5].

2.2.1. Code First pristup

Code First pristup za kreiranje strukture podataka je koristan u DDD-u (*Domain Driven Design*). Ovaj pristup omogućava programerima da se usredsrede na dizajn domena i da kreiraju klase u skladu sa zahtevima, a ne na osnovu same baze podataka [6]. Time programer usled promene šeme baze podataka nema osećaj da se vraća unazad, a kreirane klase se uvek podudaraju sa bazom podataka čije je kreiranje zasnovano nad tim klasama. Kod *Code First* pristupa ne postoji vizuelni model i sve počinje od konceptualnog modela. Bez obzira na to da li baza podataka već postoji ili ne, potrebno je kreirati klase i njene atribute. Kako je jednostavnije i brže dodati novi atribut klasi u kodu, nego novu kolonu u tabeli, *Code First* pristup to sve automatski pokriva time što na osnovu kreiranih klasa generiše migraciju uz pomoć koje se kreira šema baze podataka.

2.4. SQL Server

Da bi se razumeo pojam SQL servera, potrebno je shvatiti šta je SQL (*Structured Query Language*). SQL je programski jezik posebne namene dizajniran za rukovanje podacima u relacionom sistemu upravljanja bazama podataka. Server baze podataka je računarski program koji pruža usluge baze podataka drugim programima i računarima. Prema tome, SQL server je server baze podataka koji implentira SQL [7].

2.5. Angular

Angular predstavlja *framework* za dizajniranje i stvaranje efikasnih aplikacija na jednoj stranici (engl. *Single Page Application*) [8]. *Single page* aplikacije su aplikacije koje se izvršavaju u *browser*-u i koje ne zahtevaju ponovno učitavanje stranice za vreme interakcije korisnika sa aplikacijom, što značajno ubrzava aplikaciju i tako unapređuje korisničko iskustvo.
Angular pruža ugrađene funkcije za animaciju, HTTP (*HyperText Transfer Protocol*) uslugu i materijale koji zauzvrat imaju funkcije kao što su automatsko dovršavanje, navigacija, traka sa alatkama i menije. Kod je napisan u *TypeScript*-u koji se kompajlira u *JavaScript* i prikazuje se u pretraživaču [9].

3. ENTITETI SISTEMA

Sistem se najopštije može definisati kao skup entiteta i njihovih međusobnih veza. Entiteti predstavljaju objekte realnog sistema i oni su međusobno povezani određenim vezama. Svaki entitet ima svoje atribute koji opisuje njegove karakteristike [10]. Tako je za razvijanje sistema za automatizaciju procesa raspoređivanja zaposlenih potrebno indentifikovati njegove entitete, a to su zaposleni, veštine zaposlenih, posao, tip posla, rangiranje, ograničenje i podešavanja.

3.1. Zaposleni

Zaposleni predstavljaju najbitniji entitet ovog sistema, jer se upravo radi o njihovom raspoređivanju po poslovima. Atributi koji opisuju karakteristike zaposlenog se mogu svrstati u dve grupe i to personalni atributi i sistemski atributi.

Personalni atributi su atributi koji su direktno vezani za zaposlenog kao osobu, a sistemski atributi su atributi koji su dodati zaposlenom njegovom egzistencijom u sistemu. Prema tome, personalni atributi zaposlenog su ime, prezime, *e-mail* adresa, datum zaposlenja, datum prestanka radnog odnosa, pol, broj telefona, datum rođenja, mesto u kome stanuje, država u kojoj stanuje, a sistemski atributi su korisničko ime i lozinka.

3.2. Veštine zaposlenih

Veštine se mogu podeliti na veštine koje su zasnovane na stručnom znanju i prenosive veštine odnosno veštine koje se stiču čitavog života. U ovom sistemu veštine nisu ograničene ni na jednu grupu, već je moguće dodati bilo koju veštinu, a nju opisuje samo njen naziv.

3.3. Posao

Posao je entitet sistema koji opisuje određeni zadatak zaposlenih. Najbitnija obeležja posla u ovom sistemu su datum i vreme početka i datum i vreme završetka. Pored toga, posao dodatno opisuju naziv, tip posla i trajanje posla.

3.4. Tip posla

Tip posla omogućava svrstavanje poslova u određene kategorije kako bi se sa njima lakše upravljalo. Na primer, tip posla u nekom sistemu može da bude administracija.

3.5. Rangiranje

U sistemima koji upravljaju zaposlenima, zaposleni su sortirani obično po imenu i prezimenu. U ovom sistemu rangiranje zaposlenih na poslu podrazumeva zapravo njihovo sortiranje. Ovo sortiranje se vrši po unapred definisanim pravilima. Rangiranje zaposlenih opisuje njegov naziv, vremenski opseg i tip rangiranja.

3.6. Ograničenja

Ograničenje predstavlja entitet na osnovu kog sistem odlučuje da li zaposleni može da bude dodeljen nekom poslu ili ne. Ograničenja u ovom sistemu su unapred definisana, a odlikuje ih njihov naziv.

3.7. E-Mail podešavanje

E-Mail podešavanje predstavlja entitet koji služi za konfiguraciju slanja *e-mail*-ova korisnicima aplikacije. Obeležja koja karakterišu ovo podešavanje su SMTP

(*Simple Mail Transfer Protocol*) server, SMTP port, *e-mail* adresa i lozinka.

4. OPIS FUNKCIONALNOSTI SISTEMA

Funkcionalnosti odabranog sistema za automatizaciju procesa raspoređivanja zaposlenih se mogu svrstati u tri grupe, a to su upravljanje podacima o zaposlenima, upravljanje podacima o poslovima i upravljanje korisničkim nalozima.

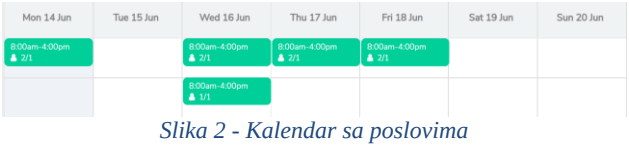
4.1. Upravljanje podacima o zaposlenima

Administrator sistema može da izvršava sve CRUD operacije nad zaposlenima popunjavajući vrednosti njihovih personalnih atributa, a pored toga ima permisiju da ažurira i sistemske podatke – korisničko ime i lozinku, u slučaju da zaposleni zaboravi iste. Zaposleni može da ažurira svoje podatke i pregleda podatke drugih zaposlenih. Administrator sistema, koji se ujedno posmatra i kao menadžer, može da dodaje veštine zaposlenom. On to radi označavanjem elementa matrice gde kolona predstavlja željenu veštinu, a red predstavlja ime i prezime zaposlenog kome se zadaje određena veština. To koje će se kolone nalaziti u matrici zavisi od toga koje veštine su dodate u sistem. Primer je prikazan na Slika 1.

Slika 1 - Matrica veština zaposlenih

4.2. Upravljanje podacima o poslovima

Menadžer može da kreira poslove koji se prikazuju na kalendaru. Prilikom kreiranja posla prikazuje se lista sa zaposlenima koji mogu biti angažovani na tom poslu. Menadžer može da angažuje dostupne zaposlene na određenom poslu. U slučaju da to uradi, svakom zaposlenom će na kalendaru biti prikazan posao na kome je on angažovan. Primer kalendara na kome se nalaze poslovi je prikazan na Slika 2.



Slika 2 - Kalendar sa poslovima

4.2.1. Ograničenja

Angažovanje zaposlenog je zavisno od ograničenja koja su definisana unutar sistema. Ta ograničenja sistema su unapred definisana i ne mogu se menjati, a to su „*Zaposleni nije angažovan*“, „*Zaposleni odlazi*“, „*Zaposleni je počeo da radi*“ i „*Zaposleni zadovoljava potrebne veštine*“.

„Zaposleni još uvek nije angažovan“ ograničenje je ograničenje za angažman zaposlenog gde se obezbeđuje da zaposleni ne može u isto vreme da radi na dva posla. U slučaju da menadžer pokuša da angažuje zaposlenog u vreme kad je on već angažovan, sistem će mu izbaciti poruku da ne može da izvrši tu akciju. Međutim, poslovi unutar nekog sistema ne moraju biti poslovi koji se obavljaju odjednom u celini.

Prema tome, posao se može posmatrati i kao određeni zadatak (engl. *Task*), pa postoji izuzetak gde je u redu da zaposleni radi na jednom zadatku i odmah po završetku nastavi da radi na sledećem zadatku. Na primer, ako zadatak traje od 8 do 16 časova, moguće je kreirati zadatak koji traje od 16 do 17 časova. Ovo se može primeniti kod posla kao što je kreiranje nekog izveštaja gde nakon završetka kreiranja izveštaja zaposleni štampa i dostavlja izveštaj.

„Zaposleni odlazi“ ograničenje je ograničenje koje se odnosi na angažovanje zaposlenog u slučaju da zaposleni odlazi iz kompanije. Ovo ograničenje se ne odnosi samo na period nakon datuma završetka, već i na sam datum završetka. Razlog je što postoji određeni rizik zadavanja novog zadatka zaposlenom, jer postoji mogućnost da zaposleni ne uspe za određeno vreme da obavi svoj posao.

„Zaposleni počinje da radi“ ograničenje je logičko ograničenje, više kao neka vrsta validacije da se zaposleni ne može angažovati na poslu, ako on nije ni počeo da radi u kompaniji.

4.2.2. Rangiranja

Pored ograničenja, ovaj sistem ima prethodno pomenuto rangiranje koje služi za sortiranje zaposlenih unutar liste dostupnih zaposlenih kako bi na vrhu bili zaposleni koji bi, na osnovu unapred definisanih pravila, bili najbolji izbor za angažovanje na tom poslu. Najbolji izbor u smislu da zaposleni ne budu preopterećeni poslom. Ova rangiranja se mogu konfigurisati unutar sistema, a to su „*Učestalost dana u nedelji*“ i „*Raspored sati*“.

„*Učestalost dana u nedelji*“ predstavlja sortiranje zaposlenih po broju angažmana koja su se desila na isti dan u nedelji. Na primer, ako postoji definisano rangiranje gde se zaposleni sortiraju po angažmanima ponedeljkom na mesečnom nivou i ukoliko je neki zaposleni već raspoređen na poslove koji su ponedeljkom, sistem će „potencirati“ da neki drugi zaposleni koji je imao manje angažmana ponedeljkom bude angažovan na tom poslu, koji takođe pada na ponedeljak. Pošto poslovi mogu da traju više dana, ovde se dolazi do problema da li se dan u sred posla posmatra kao inkrement unutar kalkulacija. Zbog ovog slučaja korišćenja, dodato je polje koje označava to da li će ti dani predstavljati inkrement ili ne.

Na sličan način funkcioniše i drugo rangiranje – „*Raspored sati*“ koje sortira zaposlene po ukupnom broju sati trajanja angažmana u definisanom vremenskom periodu. Na primer, ukoliko je zaposleni A u jednoj nedelji bio angažovan već 20 sati, a zaposleni B je u toj istoj nedelji bio angažovan 8 sati, sistem će „potencirati“ zaposlenog B odnosno na vrhu liste dostupnih radnika biće zaposleni B, jer bi u slučaju angažovanja zaposlenog A on bio preopterećen obimom posla.

4.3. Upravljanje korisničkim nalozima

Jedna vrsta korisnika ovog sistema je administrator, odnosno menadžer koji rukuje poslovima i podacima

unutar sistema. Iza ovog korisničkog naloga može se nalaziti više menadžera. Sa druge strane, zaposleni su takođe korisnici ovog sistema koji imaju svoje korisničko ime i lozinku uz pomoć koje pristupaju sistemu.

Administrator sistema drugim korisnicima može da podesi lozinku, a oni prilikom prvog prijavljivanja moraju izmeniti tu lozinku i ulogovati se novom lozinkom. Zaposleni unutar svog profila u već pomenutom delu za sistemske podatke, mogu da promene svoju lozinku. U slučaju da neki korisnik zaboravi svoju lozinku, potrebno je da unese svoju *e-mail adresu*, nakon čega će mu biti poslat *e-mail* sa novom slučajno generisanom lozinkom pomoću koje na stranici za prijavljivanje inicira ponovno postavljanje lozinke. *E-mail* adresa sa koje će biti poslata poruka za promenu lozinke je *e-mail* adresa koja je podešena u *e-mail* podešavanjima unutar aplikacije.

5. ZAKLJUČAK

Stalne tehnološke i tržišne promene postavljaju sve veće zahteve pred menadžere. U takvim uslovima, razumno i delotvorno upravljanje resursima temelj je svakog uspešnog poslovanja [11]. Mnoge kompanije zahtevaju fleksibilan pristup planiranju koji zavisi od neprestanih promena u oblasti kojom se kompanija bavi. Da ne bi dolazilo do propusta i negativnih iznenađenja, osmišljeni su sistemi za automatizaciju procesa raspoređivanja zaposlenih. Ovi sistemi utiču na unapređenje procesa rada i bolju organizaciju unutar kompanije koja koristi taj sistem. Da ovi sistemi ne bi predstavljali samo modernu zamenu *Excel* tabele, dodatno su osmišljene funkcionalnosti koje olakšavaju rad menadžera prilikom raspoređivanja zaposlenih čime se osigurava to da zaposleni ne budu opterećeni prevelikim obimom posla. Odabirom zaposlenog koji je, na osnovu unapred definisanih pravila, najkompatibilniji za rad na određenom poslu osigurava se i poslovni uspeh.

U ovom radu opisan je sistem koji predstavlja pomoć menadžerima pri raspoređivanju zaposlenih po poslovima. On u osnovi predstavlja raspored rada kompanije i podložan je svakodnevnim promenama u zavisnosti od potreba. Ovo IT rešenje je proširivo tj. mogu se dodavati nove funkcionalnosti poput novih ograničenja i rangiranja koja olakšavaju posao i brinu o raspoređivanju zaposlenih.

6. LITERATURA

[1] Institute Of Knowledge Management, Knowledge International Journal Scientific and applicative papers V-4-, Skoplje, 2014.

[2] P. B. Dimitrijević, Planiranje ljudskih resursa, regrutovanje i selekcija kandidata. Dostupno na: https://www.seminarski-diplomski.co.rs/LJUDSKI%20RESURSI/LjudskiResursi.htm. [Poslednji pristup 15. jun 2021.]

[3] L. Martin, „Everything You Need To Know About Employee Management (Tips + Tools),“ Time Doctor. Dostupno na: https://biz30.timedoctor.com/employee-management/. [Poslednji pristup 14. jun 2021.]

[4] D. Roth, R. Anderson i S. Luttin, „Introduction to ASP.NET Core,“ Microsoft, 17. april 2020. Dostupno na: https://docs.microsoft.com/en-us/aspnet/core/introduction-to-aspnet-core?view=aspnetcore-5.0. [Poslednji pristup 13. jun 2021.]

[5] B. Kolar i V. Blagojević, „ENTITY FRAMEWORK za data-orijentisane aplikacije,“ HelloWorld.rs, 27. mart 2015. Dostupno na: https://www.helloworld.rs/blog/ENTITY-FRAMEWORK-za-data-orijentisane-aplikacije/289. [Poslednji pristup 13. jun 2021.]

[6] Y. Li, R. Gao, C. C. Xiaobin Kang i Q. Zhou, A watersheddata management and visualization system using code-first approach, New York: Springer Science+Business Media , 2016.

[7] E. Burns, „What is SQL Server?,“ Tech Monitor, 16. februar 2017. Dostupno na: https://techmonitor.ai/what-is/what-is-sql-server-4914415. [Poslednji pristup 13. jun 2021.]

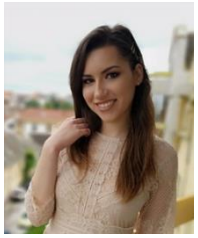
[8] „Introduction to the Angular Docs“ Dostupno na: https://angular.io/docs. [Poslednji pristup 13. jun 2021.]

[9] „Angular7 Tutorial,“ Tutorials Point. Dostupno na: https://www.tutorialspoint.com/angular7/index.htm. [Poslednji pristup 13. jun 2021.]

[10] „Semantičko modeliranje,“ 2010. Dostupno na: http://poincare.matf.bg.ac.rs/~ivana/courses/pbp/pbp.cas1.SemantickoModeliranje.pdf. [Poslednji pristup 13. jun 2021.].

[11] „Ljudski resursi - Upravljanje ljudskim resursima“ Dostupno na: https://www.seminarski-diplomski.co.rs/LJUDSKI%20RESURSI/LjudskiResursi.htm. [Poslednji pristup 15. jun 2021.]

KRATKA BIOGRAFIJA



Zorica Babić rođena je 7. avgusta 1996. godine u Somboru. Kao nosilac Vukove diplome završila je Osnovnu školu „Žarko Zrenjanin“ u Apatinu, a zatim Srednju ekonomsku školu u Somboru gde je pohađala smer finansijski administrator. 2015. godine je upisala inženjerstvo informacionih sistema na Fakultetu tehničkih nauka u Novom Sadu gde je jula 2019. odbranila diplomski rad i zatim upisala master studije iz iste oblasti.

U realizaciji Zbornika radova Fakulteta tehničkih nauka u toku 2020. godine učestvovali su sledeći recenzenti:

Aco Antić
Aleksandar Erdeljan
Aleksandar Kovačević
Aleksandar Kupusinac
Aleksandar Ristić
Bato Kamberović
Biljana Njegovan
Bogdan Kuzmanović
Bojan Batinić
Bojan Lalić
Bojan Tepavčević
Bojana Beronja
Branislav Atlagić
Branislav Nerandžić
Branka Nakomčić
Branko Milosavljević
Branko Škorić
Damir Đaković
Danijela Ćirić
Danijela Gračanin
Danijela Lalić
Darko Čapko
Darko Marčetić
Darko Reba
Dejan Ecet
Dejan Jerkan
Dejan Ubavin
Dejana Nedučin
Dragan Ivanović
Dragan Jovanović
Dragan Ivetić
Dragan Jovanović
Dragan Kukolj
Dragan Mrkšić
Dragan Pejić
Dragan Šešlija
Dragana Bajić
Dragana Konstantinović
Dragana Šarac
Dragana Štrbac
Dragoljub Šević
Dubravka Bojanić
Dušan Dobromirov
Dušan Gvozdenac
Dušan Kovačević
Dušan Uzelac
Duško Bekut
Đorđe Ćosić
Đorđe Lađinović
Đorđe Obradović
Đorđe Vukelić
Đula Fabian
Đura Oros
Đurđica Stojanović
Filip Kulić
Goran Sladić
Goran Švenda
Gordana Milosavljević
Gordana Ostojić
Igor Budak
Igor Dejanović
Igor Karlović
Igor Peško
Ivan Beker
Igor Maraš
Ivan Mezei
Ivan Todorović
Ivana Katić
Ivana Kovačić
Ivana Maraš
Ivana Miškeljin
Jasmina Dražić
Jelena Atanacković
Jeličić
Jelena Borocki
Jelena Demko Rihter
Jelena Radonić
Jelena Slivka
Jelena Spajić
Jovan Petrović
Jovanka Pantović
Laslo Nađ
Lazar Kovačević
Leposava Grubić Nešić
Livija Cvetičanin
Ljiljana Vukajlov
Ljiljana Cvetković
Ljubica Duđak
Maja Turk Sekulić
Marinko Maslarić
Marko Marković
Marko Todorov
Marko Vekić
Maša Bukurov
Matija Stipić
Milan Čeliković
Milan Mirković
Milan Rapajić
Milan Segedinac
Milan Simeunović
Milan Trifković
Milan Trivunić
Milan Vidaković
Milena Krklješ
Milica Kostreš
Milica Miličić
Mijodrag Milošević
Milovan Lazarević
Miodrag Hadžistević
Miodrag Zuković
Mirjana Damnjanović
Mirjana Malešev
Miroslava Radeka
Mirko Borisov
Miro Govedarica
Miroslav Hajduković
Miroslav Kljajić
Miroslav Popović
Miroslav Zarić
Mitar Jocanović
Mitar Đogo
Mladen Kovačević
Mladen Tomić
Mladen Radišić
Nebojša Brkljač
Neda Milić Keresteš
Nemanja Stanisavljević
Nemanja Sremčev
Nikola Đurić
Nikola Jorgovanović
Nikola Radaković
Ninoslav Zuber
Ognjen Lužanin
Pavel Kovač
Peđa Atanasković
Petar Malešev
Platon Sovilj
Predrag Šiđanin
Radivoje Dinulović
Radomir Kojić
Radovan Štulić
Relja Strezoski
Slavica Mitrović
Slavko Đurić
Slobodan Dudić
Slobodan Krnjetin
Slobodan Morača
Sonja Ristić
Srđan Kolaković
Srđan Popov
Srđan Vukmirović
Staniša Dautović
Stevan Gostojić
Stevan Milisavljević
Stevan Stankovski
Strahil Gušavac
Svetlana Bačkalić
Svetlana Nikoličić
Tanja Kočetov
Tatjana Lončar - Turukalo
Uroš Nedeljković
Valentina Basarić
Velimir Čongradec
Veran Vasić
Veselin Perović
Višnja Žugić
Vladimir Katić
Vladimir Mučenski
Vladimir Strezoski
Vlado Delić
Vlastimir Radonjanin
Vojin Ilić
Vuk Bogdanović
Zdravko Tešić
Zoran Anišić
Zoran Brujić
Zoran Čepić
Zoran Jeličić
Zoran Mitrović
Zoran Papić
Željen Trpovski
Željko Jakšić