УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

ЗБОРНИК РАДОВА
ФАКУЛТЕТА ТЕХНИЧКИХ НАУКА

Едиција: Техничке науке - зборници
Година: XXXV
Број: 3/2020

Нови Сад

Едиција: „Техничке науке – Зборници“
Година: XXXV *Свеска*: 3

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

Месечно

ISSN 0350-428X

COBISS.SR-ID 58627591

ПРЕДГОВОР

Поштовани читаоци,

Пред вама је трећа 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 30.09.2019. до 31.10.2019. год., а који се промовишу 23.03.2020. 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 три свеске.

У овој свесци, са редним бројем 3., објављени су радови из области:

- машинства и
- електротехнике и рачунарства.

У свесци са редним бројем 4. објављени су радови из области:

- грађевинарства,
- саобраћаја,
- графичког инжењерства и дизајна,
- архитектуре и
- инжењерства информационих система.

У свесци са редним бројем 5. објављени су радови из области:

- инжењерског менаџмента,
- инжењерства заштите на раду и заштите животне средине,
- мехатронике,
- геодезије и геоматике,
- инжењерства третмана и заштите вода (TEMPUS)
- управљања ризиком од катастрофалних догађаја и пожара,
- сценске архитектуре и дизајна и
- биомедицинског инжењерства.

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

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

„Високо место у друштву најбољих“

Уредништво

SADRŽAJ

STRANA

Radovi iz oblasti: Mašinstvo

STRANA

Radovi iz oblasti: Elektrotehnika i računarstvo

STRANA

STRANA

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621
DOI: https://doi.org/10.24867/07AM02Tepavac

UNAPREĐENJE LABORATORIJSKOG MODELA MOSNE DIZALICE UGRADNJOM ELEKTROMOTORNIH POGONA DIZANJA I KRETANJA

IMPROVEMENT OF THE BRIDGE CRANE LABORATORY MODEL BY INSTALLATION OF ELECTRIC MOTOR DRIVES FOR HOISTING AND TRAVELLING

Nikola Tepavac, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj – *U okviru ovog rada opisana je izrada pogonskih mehanizama i upravljačke elektronike modela dvogredne mosne dizalice, koja se nalazi u Laboratoriji za mašinske konstrukcije, transportne i građevinske mašine Fakulteta tehničkih nauka u Novom Sadu.*

Ključne reči: *model mosne dizalice, elektromotorni pogoni, upravljanje pogonima dizalice.*

Abstract – *This paper describes the development of driving mechanisms and control electronics of the bridge crane laboratory model, located in the Laboratory for mechanical structures, materials handling and construction machines of the Faculty of Technical Sciences in Novi Sad.*

Keywords: *bridge crane model, electric motor drives, crane drives control.*

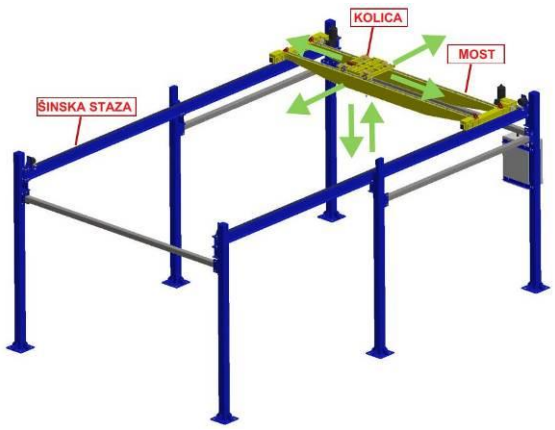
1. UVOD

Mosne dizalice se koriste pri manipulaciji najrazličitijim vrstama tereta u proizvodnim halama, radionicama, skladišnim prostorima, energetskim objektima, livnicama, kod obavljanja raznih tehnoloških procesa, montaže, demontaže opreme i slično. Tokom rada kreću se iznad podova hala, po šinama koje leže na šinskim nosačima, oslonjenim na konzolama nosećih armirano-betonskih ili čeličnih stubova. Podizanje i prenos raznih tereta u okviru unutrašnjeg transporta obavlja se pomoću pogona dizanja na pokretnim kolicima i kretanjem mosta dizalice. Manipulacioni prostor mosne dizalice je definisan rasponom mosta, visinom postavljanja šinske staze i dužinom šina.

Razmatrani laboratorijski model noseće konstrukcije dvogredne mosne dizalice prvenstveno je bio namenjen za statička ispitivanja (sl. 1) na Mašinskom fakultetu (kasnije Fakultetu tehničkih nauka) u Novom Sadu. U toj izvedbi, model dizalice nije imao pogonske mehanizme, niti upravljačku elektroniku. Noseća konstrukcija je izrađena 1970. godine u okviru magistarskog rada [2]. Ovaj laboratorijski model predstavljao je proporcionalno umanjenu konstrukciju dizalice u HE Bajina Bašta (u razmeri 1:10).

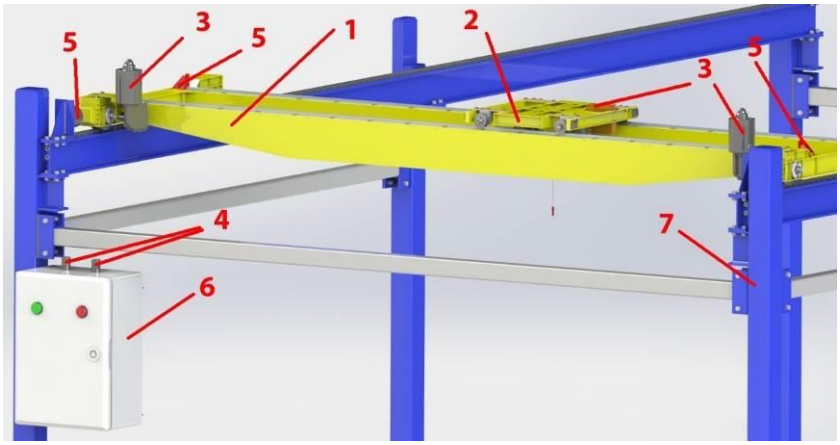
Slika 1. *Model mosne dizalice (1970), [2]*

Radna kretanja unapređenog modela mosne dizalice su dizanje/spuštanje tereta, kretanje kolica i kretanje mosta (sl. 2). Nosivost noseće konstrukcije modela je 250 kg, međutim nosivost pogona dizanja je smanjena na 10 kg (zbog čvrstoće delova mehanizma za dizanje, instalisane snage elektromotora i sl). Brzine radnih kretanja (pri radu sa nominalnim teretom) su sledeće: dizanje/spuštanje ≈ 1 m/min, kretanje mosta ≈ 6 m/min i kretanje kolica ≈ 3 m/min. Raspon dizalice je 1,84 m, dok je dužina šinske staze 3 m. Visina dizanja tereta je 1,4 m. Idejno rešenje šinske staze modela dizalice je obrađeno u zasebnom radu [1]. U međuvremenu, izrađena je šinska staza u Laboratoriji za mašinske konstrukcije, transportne i građevinske mašine na FTN, u okviru posebnog master rada i studentskog projekta Katedre za mašinske konstrukcije, transportne sisteme i logistiku.



Slika 2. *Radna kretanja modela mosne dizalice*

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Atila Zelić, docent.



Slika 3. *3D prikaz laboratorijskog modela mosne dizalice na šinskoj stazi*

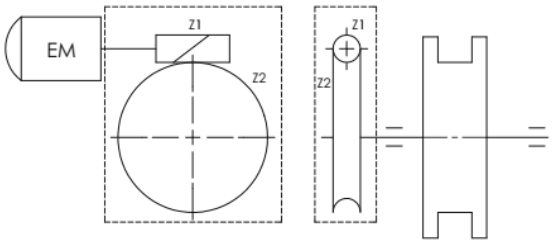
Na modelu mosne dizalice (sl. 3) mogu se uočiti određene celine: noseća konstrukcija mosta (1) i kolica (2), pogonski mehanizmi (3), električne instalacije (4), bezbednosni i zaštitni uređaji (5), elektro-ormar (6) sa visećom komandnom kutijom i konstrukcija šinske staze (7).

2. POGONSKI MEHANIZMI MODELA DIZALICE

Preliminarne koncepcije pogonskih mehanizama modela mosne dizalice prvobitno su obrađene u [3], pri čemu su uzeta u obzir kako konstrukciona, tako i materijalna ograničenja. Ova rešenja su definisana tako da ostvare potrebne funkcije, uz zadržavanje približne modelske razmere. Shodno tome, kod pogonskih mehanizama, akcenat je stavljen na kompaktnost. U nastavku su date konačne koncepcije pogonskih mehanizama modela.

2.1. Kinematske šeme pogonskih mehanizama dizalice

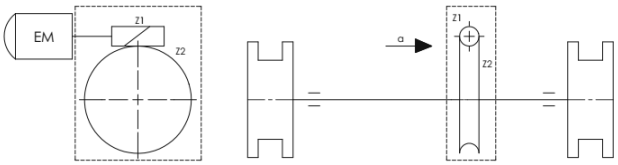
Kinematska šema pogona jedne strane mosta je data na sl. 4. Prenos kretanja od elektromotora do pogonskih točkova se ostvaruje preko pužnog reduktora.



Slika 4. *Kinematska šema pogona kretanja mosta*

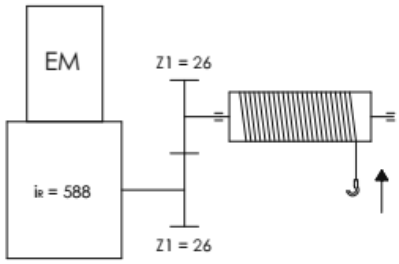
Samokočivost pužnog reduktora omogućava držanje mosta u zakočenom stanju, tako da pri transportu tereta inercijalne sile i uticaji okoline ne mogu izazvati neželjena pomeranja po šinskoj stazi.

Kinematska šema pogona kretanja kolica je data na sl. 5. Slična je kinematskoj šemi pogona mosta, ali razlika je u ugradnji centralnog vratila do točkova na kolicima.



Slika 5. *Kinematska šema pogona kretanja kolica*

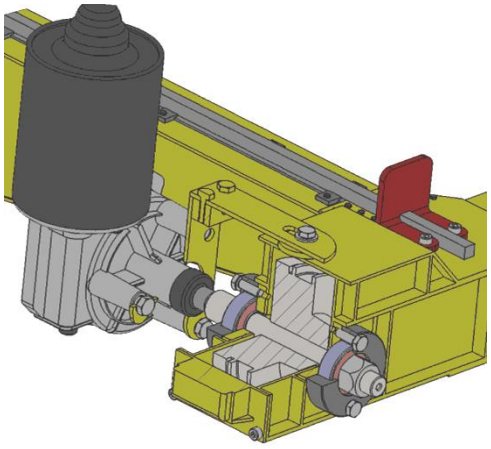
Kinematska šema pogona dizanja je prikazana na sl. 6. Elektromotor sa pužnim reduktorom prenosi obrtno kretanje na bubanj za namotavanje užeta, preko dva spregnuta zupčanika. Pošto nije potrebno vršiti dodatnu redukciju brzine obrtanja, prenosni odnos ovog zupčastog para je 1.



Slika 6. *Kinematska šema pogona dizanja*

2.2. Izvedba pogonskog mehanizma za kretanje mosta

Mesta na bočnim nosačima mosta (sl. 7), koja su predviđena za montažu kućišta ležajeva, dodatno su ojačana pločicama. Iste omogućavaju preciznu ugradnju ležajeva točkova.

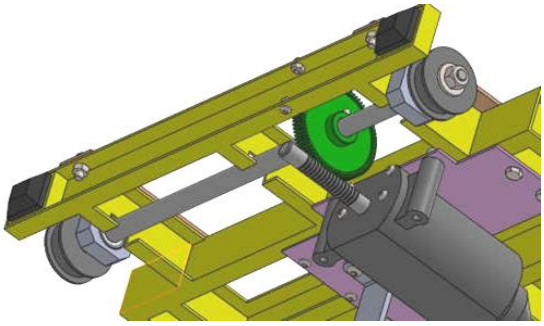


Slika 7. *Pogon kretanja jedne strane dizalice*

Vratilo ovog pogonskog mehanizma je izrađeno od čelika odgovarajućih karakteristika, sa predviđenim mestom za montažu perforiranog diska za rotacioni enkoder. Na osnovu toga, vratilo je izvedeno sa unutrašnjim navojem na jednom svom kraju, za postavljanje prethodno pomenutog perforiranog diska.

2.3. Izvedba pogonskog mehanizma za kretanje kolica

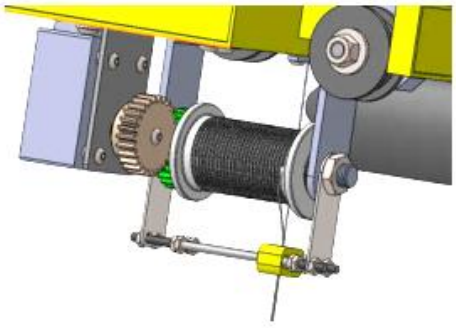
Koncepcija obuhvata jedno centralno vratila sa prenosom preko pužnog para. Pužni točak je montiran direktno na centralno vratilo, a elektromotor upravno na ovo vratilo. Vratilo se oslanja blizu svojih krajeva sa po jednim ležajem. Saosnost ovih ležajeva je ostvarena preko kućišta za ležajeve i nosećih ploča mehanizma. Kućišta ležajeva se montiraju ispod nosećih ploča koja se postavljaju na gornjoj strani kolica (sl. 8).



Slika 8. *Pogonski mehanizam za kretanje kolica*

2.4. Izvedba pogonskog mehanizma za dizanje

Ova izvedba predviđa postavljanje pogonskog mehanizma za dizanje sa donje strane roštiljne noseće konstrukcije kolica. Zbog neophodne kompaktnosti mehanizma, izlazno vratilo pužnog reduktora i osovina bubnja su postavljeni paralelno (sl. 9).



Slika 9. *Detalj pogonskog mehanizma za dizanje tereta*

3. IZRADA POGONSKIH MEHANIZAMA

Na osnovu tehničke dokumentacije, izrađene nakon koncipiranja pogonskih mehanizama, pristupilo se izradi delova za model dizalice.

Sva vratila i kućišta za ležajeve su brušeni sa kvalitetom površine Ra=0,8 μm (klasa hrapavosti N4) u h6 toleranciji kako bi se obezbedila pravilna montaža ležajeva i zupčanika. Materijal svih vratila je C45. U cilju zaštite od korozije delovi su brunirani. Na mestima montaže kućišta ležajeva, bočni nosači modela dizalice su ojačani zavarivanjem limenih ploča (od materijala S235) debljine 5 mm (sl. 10).

Pogonski i slobodni točkovi su dorađeni kako bi se sprečio silazak istih sa šina. Ovo je postignuto smanjenjem kinematskog prečnika postojećih točkova sa Ø70 mm na Ø64 mm. Na kraju pogonskog vratila mehanizma za kretanje mosta je urezan i odgovarajući unutrašnji navoj za montažu pomenutog perforiranog diska.

Slika 10. *Pogonski mehanizam mosta*

Pri montaži centralnog vratila uklonjena su pojedina ukrućenja sa prvobitne konstrukcije kolica i zavarena su nova na odgovarajućim mestima (kako bi se dobio potreban prostor za ugradnju). Nova ukrućenja su izrađena od materijala C45. Sa donje strane kolica, na ukrućenjima, izrezan je četvrtasti žljeb radi pozicioniranja vratila (sl. 11).

Slika 11. *Pogonski mehanizam za kretanje kolica (izgled sa donje strane)*

Vratilo pogonskog mehanizma kolica izrađeno je kao stepenasto vratilo, sa različitim prečnicima za ležajeve, distantne čaure i pužni zupčanik. Na kraju vratila izrađen je unutrašnji navoj za montažu perforiranog diska rotacionog enkodera.

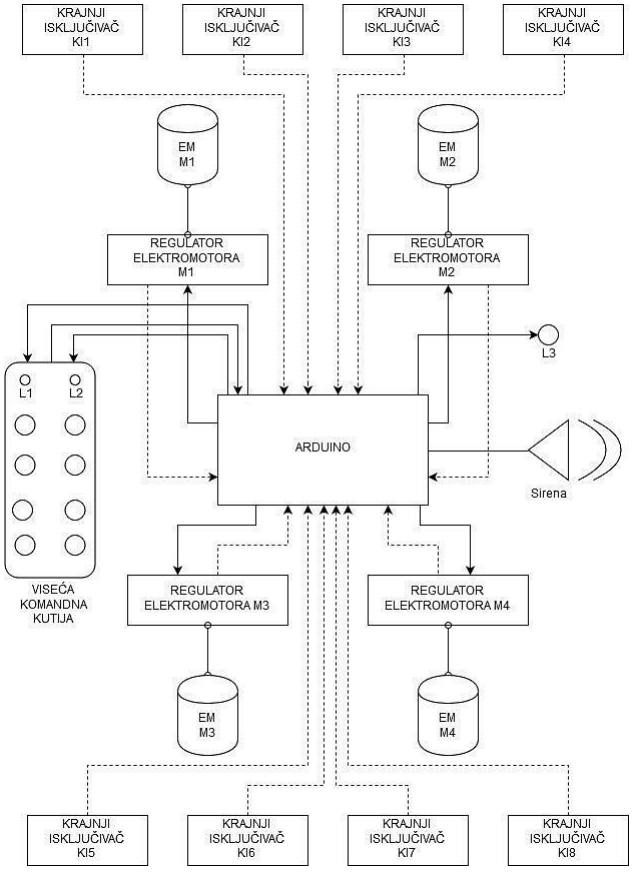
Novi točkovi kolica su izrađeni od materijala C45, sa višim vencima u odnosu na originalnu izvedbu. Aksijalno pomeranje točkova sprečeno je distantnim čaurama i samokočivim navrtkama. Osovine slobodnih točkova kolica su konzolno vezane za svoje nosače, sa distantnom čaurom između nosača i ležaja montiranog unutar točka.

Kao što je i napomenuto, pogonski mehanizam dizanja (sl. 12) se sastoji od elektromotora sa pužnim reduktorom, otvorenog zupčastog para i bubnja za namotavanje užeta. U bočni zid bubnja je upresovan gonjeni zupčanik. Isti su fiksirani i pomoću vijčane veze. Na slobodnom kraju užeta se nalazi manji teg (koji obezbeđuje zategnutost užeta i u slučaju kada dizalica radi bez tereta) i stalno zahvatno sredstvo u obliku kuke.

Slika 12. *Pogonskih mehanizam dizanja na kolicima*

4. UPRAVLJANJE RADOM DIZALICE

Rukovanje razmatranim modelom mosne dizalice je potpuno isto načinu rukovanja realnom dizalicom, putem viseće komandne kutije. Pošto je u pitanju laboratorijski model, omogućeno je i upravljanje radom dizalice, tj. podešavanje parametara ubrzanja svih elektromotornih pogona. Upravljačka šema modela dizalice je prikazana na sl. 13.



Slika 13. *Upravljačka šema modela dizalice*

Upravljačku jedinicu sačinjavaju regulatori rada elektromotora VNH2SP30DC (drajveri), i *Arduino MEGA* 2560 mikroupravljač. Ostvarena je mogućnost rukovanja dizalicom putem tastera na visećoj komadnoj kutiji i mogućnost softverskog podešavanja režima rada preko *Arduino IDE* razvojnog okruženja.

Za model dizalice izrađen je specijalni elektro-ormar u kojem su smeštene sve upravljačke komponente, kao i napajanje svih elektromotora. Napajanje ovog ormara je nominalnog napona 230 V AC i snage 450 W. Sa desne strane elektro-ormara izveden je standardni VGA priključak za povezivanje viseće komandne kutije.

Na vratima ormara (sl. 14) se nalaze tri indikatorske lampice: zelena za prisustvo mrežnog napona, zelena lampica za status dizalice (uključena/isključena) i crvena za aktivnu komandu. Ukoliko crvena lampica za aktivnu komandu svetli bez prekida onda je neka komanda aktivna, a ukoliko treperi onda je aktiviran „stop“ taster. U donjem desnom uglu vrata se nalazi glavni prekidač sa ključem za aktiviranje dizalice i upravljačke elektronike. Na sva četiri kraja šina staze kolica se nalaze krajnji isključivači, isto kao i na krajevima staze dizalice.

Slika 14. *Elektro-ormar i viseća komandna kutija*

Viseća komandna kutija (sl. 14) je izvedena sa 6 tastera za upravljanje pogonima kretanja, jednim tasterom za davanje zvučnog upozorenja i „stop“ tasterom. Kućište viseće komandne kutije je izrađeno putem tehnologije 3D štampe, na osnovu CAD modela.

Kako je model mosne dizalice predviđen da se koristi u edukativne svrhe, na njemu su postavljeni senzori brzine, ubrzanja i vibracija, kao i tenziometrijske merne trake za praćenje naponskih stanja u nosećoj konstrukciji dizalice.

5. ZAKLJUČAK

Ovaj laboratorijski model mosne dizalice, zajedno sa svojom šinskom stazom, predstavlja trajno interaktivno učilo u nastavi, za potrebe Katedre za mašinske konstrukcije, transportne sisteme i logistiku Fakulteta tehničkih nauka u Novom Sadu. Kao takvo nije namenjen za obavljanje drugih poslova, već služi isključivo za izučavanje raznih fenomena pri radu mosne dizalice.

Prilikom testiranja pogonskih mehanizama modela dizalice nisu utvrđeni nedostaci i problemi u radu. Upravljačka elektronika je napravljena sa mogućnošću izmene i dopune novim komponentama, u skladu sa daljim planovima Katedre na polju unapređenja nastave.

6. LITERATURA

[1] S. Savić: *Podloge za proračun noseće konstrukcije šinske staze laboratorijskog modela mosne dizalice“*, diplomski rad, FTN, Novi Sad, 2018.

[2] N. Babin: *Mogućnost određivanja ukupne nosivosti nosača mostovskog krana na temelju modelskog ispitivanja, statičkim simuliranjem dinamičkog opterećenja*, magistarski rad, Mašinski fakultet, Beograd, 1970.

[3] N. Tepavac: *Projekat unapređenja laboratorijskog modela mosne dizalice*, diplomski rad, FTN, Novi Sad, 2018.

Kratka biografija:

Nikola Tepavac rođen je u Kikindi 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Mehanizacija i konstrukciono mašinstvo odbranio je 2019. god. Kao student, učestovao je na takmičenjima iz robotike i projektu hibridnog električnog vozila HERMES na Departmanu za mehanizaciju i konstrukciono mašinstvo. Zaposlen je kao konstruktor alatnih mašina u Kikindi.

kontakt: simplytepi@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

УДК: 621
DOI: https://doi.org/10.24867/07AM03Crljenica

ПАРАМЕТРИЈСКО МОДЕЛОВАЊЕ ТРАКАСТОГ ТРАНСПОРТЕРА

PARAMETRIC MODELING OF BELT CONVEYOR

Горан Црљеница, Драган Живанић, *Факултет техничких наука, Нови Сад*

Област – МАШИНСТВО

Кратак садржај – *У овом раду описан је поступак параметријског моделовања тракастог транспортера у програмском пакету „Catia V5-6R2016”. У првом делу је описан општи поступак параметризације, док је у другом делу, на основу претходног прорачуна, моделиран тракасти транспортер у складу са правилима параметризовања.*

Кључне речи: *Параметријско моделовање, CATIA, тракасти транспортер*

Abstract – *This paper describes the process of parametric modeling of a belt conveyor in the software package „Catia V5-6R2016”. The first part describes the general parametrization procedure, while the second part describes the belt conveyor based on the previous calculation in accordance with the parametrization rules.*

Key words: *Parametric modeling, CATIA, belt conveyor*

1. УВОД

Тракасти транспортери су транспортни уређаји чија је основна намена да пренесу робу између две тачке помоћу бескрајне траке [1]. Транспортна трака је пребачена и затегнута обично између два бубња и његови основни саставни делови су поред бескрајне траке: рамови носеће конструкције, ваљци или клизна подлога за ношење траке, затезна и погонска станица [1].

Тракасти транспортери имају веома широку примену, и обезбеђују рационалан транспорт великих количина расуте и комадне робе на великим и малим дистанцама. У овом раду узети су подаци тракастог транспортера за транспорт каменог угља, слика 1.

Слика 1. *Тракасти транспортер [2]*

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији ментор је био др Драган Живанић, ванр. проф.

2. МОДЕЛОВАЊЕ ТРАКАСТОГ ТРАНСПОРТЕРА

Помоћу програмског пакета „*Catia V5-6R2016*” извршено је моделовање тракастог транспортера. *3D* модел је израђен коришћењем следећих модула: *Part, Assembly, Shape design.* Сви делови су израђени у *Part design* модулу користећи и неке од неке од команди из *Shape design-a* ради поједностављења поступка параметризовања делова транспортера.

Први корак је моделовање хоризонталног *C* профила праве секције транспортера. Усвојен је профил *UNP100x50x6mm.* При самом моделовању битно је узети у обзир да ће овај модел бити повезан са формулама и да ће бити мењан у складу са захтевима, па је зато битно од почетка повезивати све параметре у складу са очекиваним изменама. Након што је моделован профил, на њега су постављени отвори за вертикалне носаче, носаче са ваљчаним слоговима и хоризонталне укрућујуће пречке. Следећи корак је моделовање вертикалних носача са плочицама и носача са ваљцима, хоризонталних и косих укрућења.

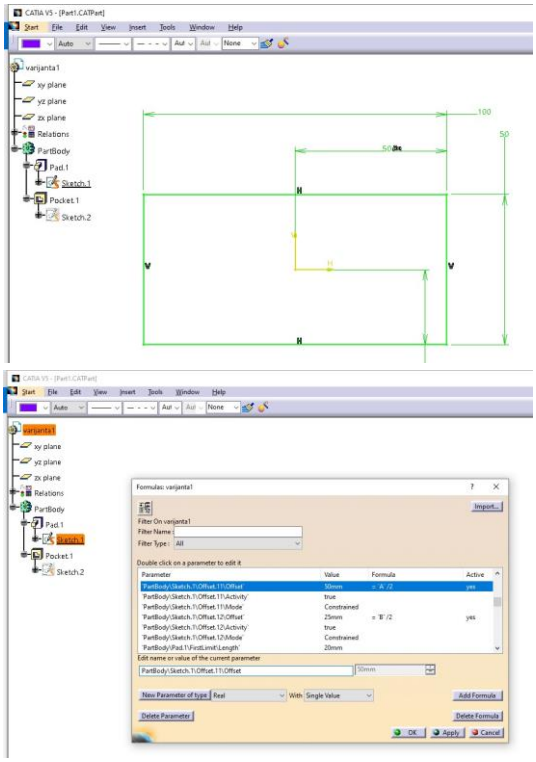
2.1. Параметризовање модела

Утврђено је да инжењери проводе највећи део свог времена на преправљање постојећих модела, док само мали проценат утроше на обликовање нових производа, док у неким случајевима та вредноста може да падне и на 0%. На основу ових података може се закључити да се баш у овом пољу мора постићи већи степен аутоматизације. Параметризација представља посебну врсту моделовања, уз коришћење формула као и *Excel* табела, при чему се смањује потребно време за моделовање и формирање база података.

Иницијално параметријско моделовање захтева значајно више времена од стандардног, али након што се добије параметризован модел промена било које вредности, тј. величине, захтева мало времена у односу на преправљање постојећег непараметри-зованог модела. Скраћење укупног потребног времена за моделовање у великој мери директно утиче на смањење финалне цене производа.

2.2. *Formula*

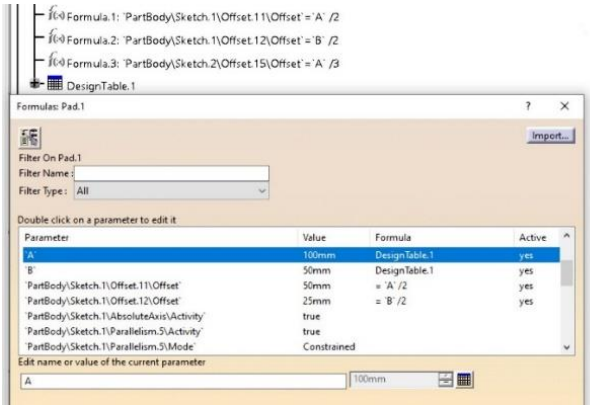
Команда *Formula* се користи за додавање веза између геометријских параметара које пројектант додаје на *3D* модел. На слици 2 се може видети поступак додавања веза између параметара. Први и основни корак је прављење *Sketch-a* у окружењу *Part design* и цртање основе модела којој се додају параметри.



Слика 2. *Команда Formula*

Након додавања формуле појавиће се симбол *f(x)* поред параметра, као ознака за његово параметризовање. У горњем левом углу у стаблу тј. у пољу *Relations*, се генеришу све унесене формуле. У било ком тренутку свака формула се може преправити и променити по потреби. Двокликом на формулу се може модификовати вредност уколико постоји потреба за тиме.

Име параметара могуће је променити тако што се обележи параметар којем је потребно променити име, отвара се команда *Formula* и обележава се име параметара које је аутоматски генерисано, а уместо њега се уписује ново име, што се може видети са слике 3. Такође, све претходно креиране формуле у пољу *Relations* ће аутоматски променити назив параметра у складу са новим заданим именом.



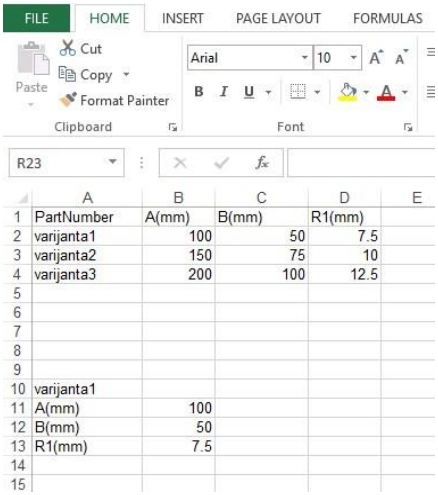
Слика 3. *Промена имена параметара*

Сви преименовани параметри могу се видети тако што се отвори команда *Formula* и у пољу *Filter type* се промени вредност на *Renamed parametars*. На листи ће се појавити сви параметри који су преименовани у том делу.

2.3. *Design Table*

Design table се може направити користећи *CATIA* датотеке, тако што се испишу параметри у *Design table* којима се задају одговарајуће вредности. Други начин за извођење овога је прављење табеле у *Microsoft Excel-u*, након чега се *Catia* повеже са *Design tabele*. Пре тога је потребно преименовати све параметре у складу са именима која су уписана у *Excel-u* због повезивања параметара.

У *Excel* се уносе имена параметара, док се у заграду уносе одговарајуће јединице за тај параметар. Јединице параметара које се користе у овом раду су дужина – *length (mm)* и угао – *angle (deg).* Након што су унете све вредности параметара за једну варијанту, уносе се вредности и за остале варијанте. Не постоји ограничење колико параметара и варијанти се могу унети у *Excel.*

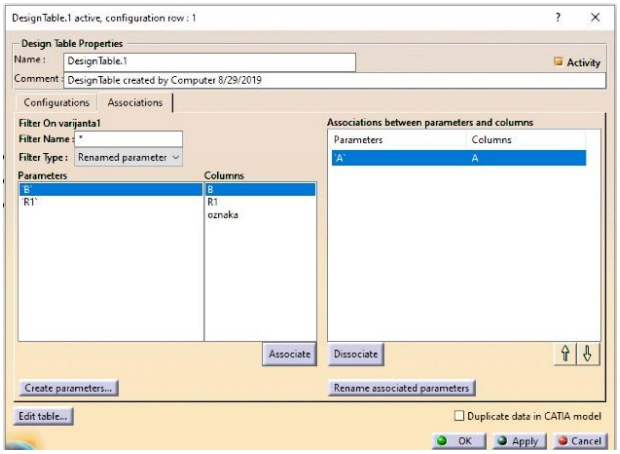


	A	B	C	D	E
1	PartNumber	A(mm)	B(mm)	R1(mm)	
2	varijanta1	100	50	7.5	
3	varijanta2	150	75	10	
4	varijanta3	200	100	12.5	
5					
6					
7					
8					
9					
10	varijanta1				
11	A(mm)	100			
12	B(mm)	50			
13	R1(mm)	7.5			
14					
15					

Слика 4. *Пример Excel табеле*

Након што су унесени сви параметри за све варијанте, креира се падајући мени помоћу кога се бира варијанта која се жели активирати. На *Sheet 2.* се уносе параметри који су изабрани падајућим менијем варијанти, јер уколико би *Design table* повезали са *Excelom, Catia* би препознала вредности параметара за варијанту 1 и након промене варијанте не би дошло до промене параметара.

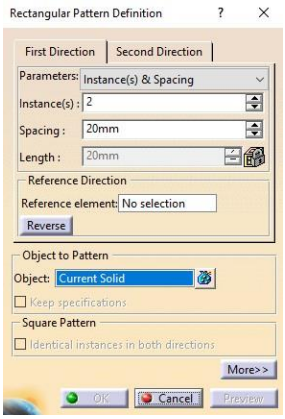
Уколико се јави потреба, вредности параметара могу се модификовати у *Excel-u* и након што се сачува, аутоматски ће доћи до промене параметара у CATIA-i.



Слика 5. *Повезивање са Design table*

3. ПАРАМЕТРИЗАЦИЈА ТРАКАСТОГ ТРАНСПОРТЕРА

Након што је завршено моделовање *C* профила праве секције почиње се са параметризацијом. Дужина самог носача повезана је са параметром из *Excel* табеле *Lp1*. Број отвора на носачу мора да зависи од дужине самог носача и да се у складу са том димензијом мења. Сви отвори су умножени командом *RecPattern* у њој су коришћене две променљиве *Instance(s)* – број понављања и *Spacing* – размак између отвора.

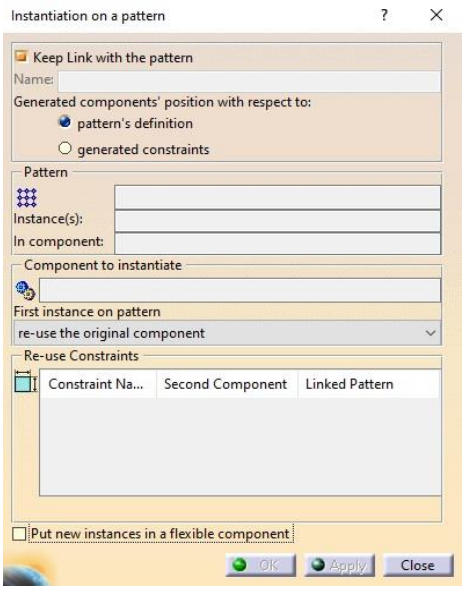


Слика 6. *Команда RecPattern*

Размак између отвора се узима на основу препорука [3] за праву секцију тј. за конкавну и конвексну кривину.

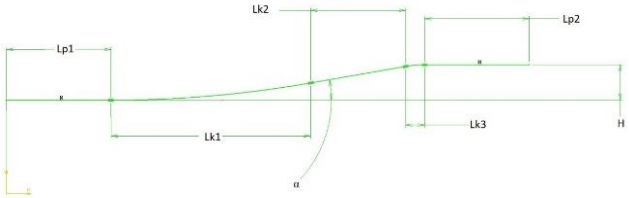
Након што је моделовани подсклоп ваљчаног слога постављен на отворе на *C* профилу, користи се команда *Reuse Pattern* за умножавање подсклопова дуж профила. Умножавање се врши тако што се у поље *Component to instantiate* одабере који се део умножава, а у поље *Pattern* се бира *Pattern* који је коришћен за умножавање отвора дуж профила што се може видети са слике 7.

На тај начин је обезбеђено да уколико дође до промене димензија, односно параметара из *Excel* табеле, број отвора, број ваљчаних слогова и вертикалних носача ће се променити.



Слика 7. *Команда Reuse pattern*

Моделирање кривина се мора извести на другачији начин него код правих секција, због промењиве геометрије и углова који хоризонтални носачи морају међусобно да заклапају. На слици 8. је приказана путања косе секције транспортера.



Слика 8. *Путања косе секције транспортера*

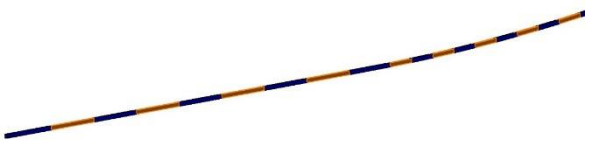
Командом *Extract* су извучене појединачне линије ових секција које се после користе за умножавање цевастих носача. Помоћу команде *Extrude ce* извлачи цео профил дуж *X* осе. Почетна тачка се поставља у нулту тачку линије профила, а на њу се постављају две линије које се користе као референце за прављење координатног система.

Прва линија је тангентна на профил, док је друга нормална на извучену површину. Координатни систем се поставља у почетну тачку, док је референца за *X* осу линија која је тангентна на профил косе секције. За *Y* осу је узета линија која је нормална на извучену површину. Профил цевастог профила се позиционира на раван координатног система.

Основна идеја параметријског моделовања ових носача је да се направи само један део и да се умножи дуж предвиђене путање. Сваки носач мора да се наслања на следећи носач под одговарајућим углом да би се испоштовала претходно моделована путања профила.

На претходно извучене линије секција, командом *Points Repetition* умножавају се тачке и равни које су нормалне на линије профила. Ове равни се користе за постављање профила *Sketch-a* носача и за сечење истих. При коришћењу команде *Points Repetition* у пољу *Curve* се означава линија дуж које ће се умножавати тачке и на коју ће равни бити нормално постављене. У поље *Spacing* се уноси растојање које дефинише дужину цевастих носача, односно међусобно растојање равни којим ће се сећи носачи.

Када су генерисане равни и тачке дуж путање, постављају се линије које повезују тачке, односно средине генерисаних равни. Ове линије се користе као правци извлачења носача дуж путање, тако што се у *Pad-u* у поље *Reference* означи ова линија. На слици 9. се може видети приказ генерисаних носача коришћењем параметријског моделовања.

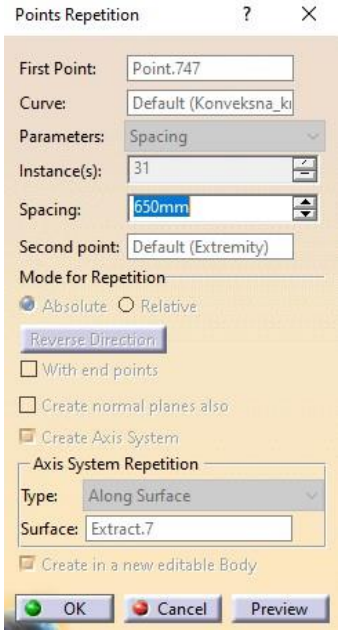


Слика 9. *Генерисани носачи косе секције*

Код конвексне и конкавне кривине потребно је додати координатне системе чија је једна оса тангентна са

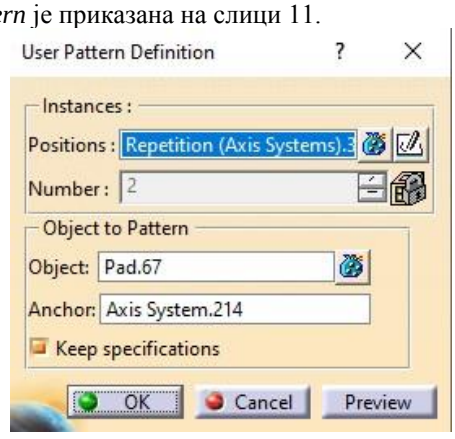
извученом линијом профила, а друга оса нормална на површину, јер вијци које је потребно додати морају бити нормални на површину цевастог носача тј. површину путање. Ово се постиже тако што се предходно моделовани координатни систем умножи помоћу команде *Points Repetition* која се може видети на слици 10.

У поље Curve се бира линија кривине, и уписује се растојање које одговара броју носача ваљака који се налазе у тој секцији транспортера. У пољу *Axis System Repetition* под *Type* бира се опција *Along Surface* тј. извучена површина путање и као референца се узима извучена површина цевастих носача.



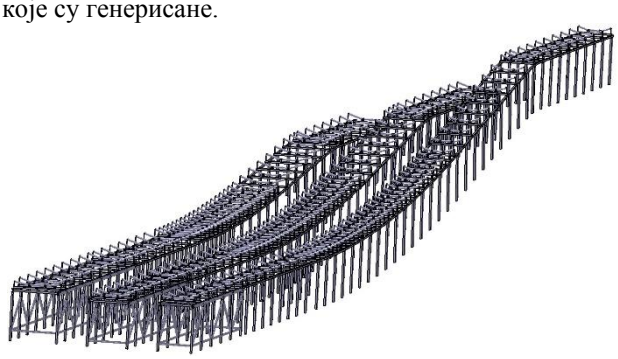
Слика 10. *Команда Points Repetition*

У косој секцији транспортера због променљиве геометрије носача није било могуће користити команду *Rectangular Pattern* која умножава дуж праве линије, већ команду *User pattern*. У поље *Object* прво се означава део који се умножава, док у поље *Positions* се бира предходно направљен *Points Repetition Axis systems* на основу кога се аутоматски дефинише број умножавања. У пољу *Anchorn* бира се координатни систем првог вертикалног носача који служи као референтни координатни систем за умножавање вертикалних носача. Команда *User pattern* је приказана на слици 11.



Слика 11. *Команда User pattern*

Након што су сви делови умножени и позиционирани на одговарајуће место у склопу, променом варијанте у *Excel* табели долази до промене *3D* модела у *Catii*. На слици 12. се могу видети резултати све три варијанте које су генерисане.



Слика 12. *Приказ варијанти тракастог транспортера*

4. ЗАКЉУЧАК

Параметријско моделовање омогућава брзе и једноставне промене параметара, што уједно смањује потребно време за израду као и време за проверу након измена, да би се утврдило да ли је дошло до колизије или неправилности елемената склопа. Аутоматизација овог процеса смањује финалне трошкове, као и могућност за појаву грешака приликом промене неког од параметара.

На основу добијених параметризованих модела тракастог транспортера може се закључити да овај вид моделовања представља будућност израде *3D* модела, због својих одличних карактеристика и могућности које пружају. Иницијално, параметријско моделирање захтева више труда, али уштеде које се остварују по питању времена, умногоме превазилазе сав додатни напор који се улаже при почетном моделовању.

5. ЛИТЕРАТУРА

[1] https://nastava.sf.bg.ac.rs/pluginfile.php/8450/mod_resource/content/0/Predavanja_kontinualna_sredstva/TrakastiTransporter.pdf (приступљено у септембру 2019.)

[2] *https://www.pinterest.com/pin/496521927646566724/*http://definicije.blogspot.rs/2013/04/ (приступљено у септембру 2019.)

[3] Механизација претовара, Јован Владић, Нови Сад, 1991.,

Кратка биографија:

Горан Црљеница рођен је у Новом Саду 1991. Дипломирао на Факултету техничких наука, 2015. године, на студијском програму Механизације и конструкционог машинства.

Драган Живанић рођен је у Сремској Митровици 1972. год. Докторирао је 2012. год, а од 2019. ради као ванр. проф. на Факултету техничких наука у Новом Саду.

Зборник радова Факултета техничких наука, Нови Сад

UDK: 621.7
DOI: https://doi.org/10.24867/07AM04Ilkic

ПРОЈЕКАТ ХОРИЗОНТАЛНОГ ТРАКАСТОГ ТРАНСПОРТЕРА

PROJECT OF HORIZONTAL BELT CONVEYOR

Слободан Илкић, Драган Живанић, *Факултет техничких наука*, Нови Сад

Област – МАШИНСТВО

Кратак садржај – *Овај рад приказује прорачун тракастог транспортера са дефинисаним основним параметрима. Рад обухвата прорачун капацитета, одређивање сила у траци, као и проверу претходно усвојених елемената транспортера. Последњи део се бави питањима безбедности при раду на овом транспортеру.*

Кључне речи: *Непрекидни транспорт, тракасти транспортер, прорачун.*

Abstract – *This paper presents calculations of a belt conveyor with given basic parameters. The paper includes capacity calculation, belt forces calculation, as well as verification of previously adopted elements. The last part concentrates on safety during working process of this conveyor.*

Keywords: *Continuous tranport, belt conveyor, calculations.*

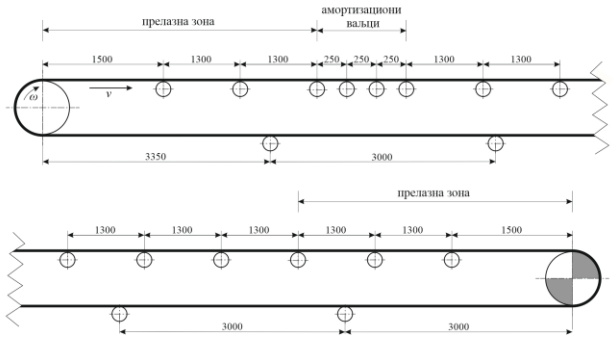
1. UVOD

Задати су основни параметри транспортера, а то су: дужина транспортера, технички капацитет, врста материјала који се транспортује и услови рада.

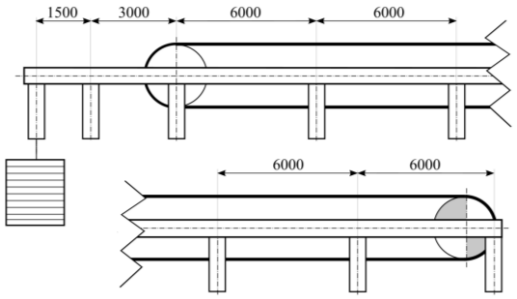
Даље у раду се наводе сви елементи који су усвојени (трака, ваљци, бубњеви, механизам за затезање, моторедуктор, утоварни уређај) и њихов распоред (слика 1).

На крају поглавља дата је и скица носеће конструције са положајима ослонаца (слика 2), као и димензије усвојеног *C* профила.

Ови подаци биће неопходни за даљи прорачун.



Слика 1. *Распоред елемената транспотера*



Слика 2. *Носећа конструкција*

2. ПРОРАЧУН КАПАЦИТЕТА И ОДРЕЂИВАЊЕ СИЛА У ТРАЦИ

2.1. Капацитет транспортера

На основу номиналног броја обртаја мотора ($n_{em(n)}$), преносног односа редуктора (i_r) и пречника бубња (d_b) долази се до брзине траке:

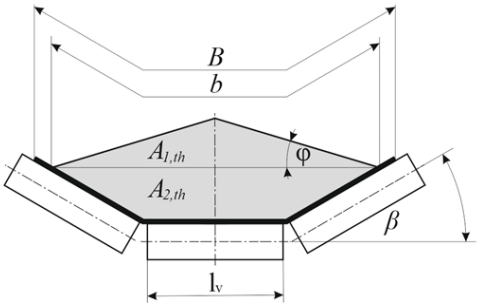
$$v = \frac{n_{em(n)} \cdot \pi \cdot d_b}{i_r} \tag{1}$$

На основу познатих димензија (слика 3) одређује се површина попречног пресека помоћу следећих израза [2]:

$$A_{1,th} = [l_v + (b - l_v) \cdot \cos\beta]^2 \cdot \frac{\tan\varphi}{4} \tag{2}$$

$$A_{2,th} = \left[l_v + \frac{(b - l_v)}{2} \cdot \cos\beta \right] \cdot \frac{(b - l_v)}{2} \cdot \sin\beta \tag{3}$$

$$A_{th} = A_{1,th} + A_{2,th} \tag{4}$$



Слика 3. *Попречни пресек материјала*

Коначно, до техничког капацитета долази се преко брзине траке (v), густине материјала (ρ), као и површине попречног пресека материјала (A_{th}):

$$Q = v \cdot \rho \cdot A_{th} \cdot k_3 \tag{5}$$

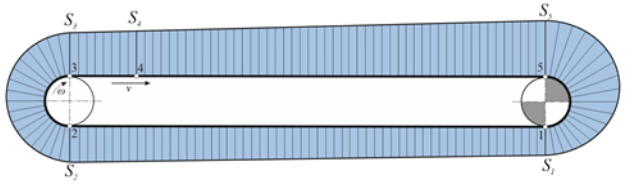
НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Драган Живанић, ванр. проф.

2.1. Силе у траци

Да би се одредиле силе у траци, прво је потребно израчунати отпоре кретања одређених елемената, а то су: отпори ваљака, отпор затезног бубња и отпор утоварног уређаја.

Након што су отпори познати, одређују се силе у траци методом обиласка контуре. Почетна тачка је одабрана на месту силазног крака погонског бубња (слика 4). Од ове тачке се, у смеру кретања траке, одређују отпори на појединим деоницама.

Слика 4. *Карактеристичне тачке на транспортеру*



Слика 5. *Расподела сила у траци транспортера*

3. ПРОВЕРА ЕЛЕМЕНАТА ТРАНСПОРТЕРА

3.1. Провера траке

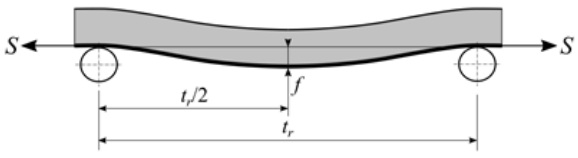
Врши се провера чврстоће а затим и провера угиба траке. Максимална дозвољена сила у траци је пропорционална јачини материјала (K), броју слојева (z) и ширини траке (B), па израз гласи [1]:

$$S_{max} = \frac{z \cdot B \cdot K}{n} \tag{6}$$

Експериментима је утврђено да је максимални дозвољени угиб траке на радној страни $f_{doz} = 0,025 \cdot t_r$ [1], јер у противном долази до претераног расипања материјала.

Из претходног се долази до минималне потребне силе у траци [1]:

$$S_{min} = 5 \cdot \left(q + q_0\right) \cdot t_r \tag{7}$$



Слика 6. *Угиб траке на радној страни*

3.2. Провера погонског бубња

Врши се провера бубња на проклизавање, затим провера електромотора и редуктора.

Провера на проклизавање се врши помоћу Ојлеровог образца [1]:

$$\varphi_e = \frac{e^{\mu\alpha} - 1}{\frac{S_5}{S_1} - 1} \geq 1,25 \tag{8}$$

Елементи који фигуришу у изразу су: коефицијент трења између бубња и траке (μ), обвојни угао (α), силе у траци у тачкама 1 и 5 (S_1 и S_5).

За проверу електромотора потребно је израчунати потребну снагу на вратилу бубња, која зависи од обимне силе коју бубањ преноси на траку (U), брзине траке (v) и коефицијента искоришћења погона бубња (η_b):

$$P_b = \frac{U \cdot v}{\eta_b} \tag{9}$$

Потребна снага електромотора је:

$$P_{em} = \frac{P_b \cdot k}{\eta_r} \tag{10}$$

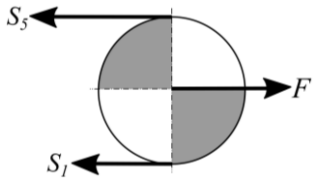
У изразу је са k означен степен сигурности, а са η_r означен коефицијент корисности редуктора.

Момент на бубњу је уствари момент на редуктору и он је једнак производу обимне силе и полупречника бубња:

$$M_b = U \cdot \frac{D}{2} \tag{11}$$

3.3. Провера погонског вратила

На бубањ делују силе од наилазног и силазног крака траке – силе S_1 и S_5 (слика 7), које се са бубња преносе на вратило.

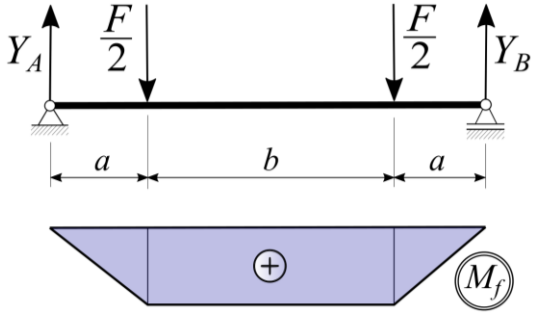


Слика 7. *Дејство сила на погонски бубањ*

Транспортер је хоризонталан и обухватни угао траке је 180°, па су и обе силе у траци хоризонталне као и сила која оптерећује вратило, и њена вредност је:

$$F = S_1 + S_5 \tag{12}$$

Вратило се посматра као греда на два ослонца, на коју делују силе на међусобном растојању b. Силе са погонског бубња се преносе на местима споја бубња са вратилом, и при том се узима у обзир да обе стране примају подједнако оптерећење (слика 8).



Слика 8. *Случај оптерећења погонског вратила*

Вредност максималног момента зависи само од растојања a (растојање од ослонца вратила на конструкцију до споја бубња и вратила – слика 8) и износи:

$$M = \frac{F}{2} \cdot a \cdot k_F \qquad (13)$$

*Напомена: Са k_F је обележен коефицијент који узима у обзир могућност неједнаке расподеле сила међу спојевима вратила и бубња.

За проверу лежајева, у овом случају примењује се динамички прорачун. Он се спроводи тако што се рачуна радни век лежаја, по образцу ([5]) :

$$L_h = \frac{10^6}{60n}\left(\frac{C}{F_L}\right)^{\alpha} \geq L_{h0} \qquad (14)$$

Елементи који фигуришу у изразу су: динамичка носивост (C), број обртаја вратила (n) и динамичко еквивалентно оптерећење лежаја (F_L).

3.4. Провера ваљака на радној страни

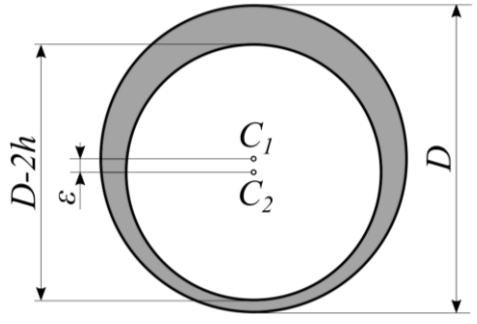
Носећи ваљци трпе оптерећење од тежине траке и од тежине материјала, а осовина трпи и додатно опеређење од тежине обртног дела ваљка.

Потребно је наћи оптерећење осовине једног ваљка. Средњи ваљак је најоптерећенији и сматраће се да прима 60% од укупне силе којом трака делује на слог. Такође, ова оптерећења су динамичког карактера, па треба узети у обзир и динамички фактор удара. Дакле, укупно оптерећење коју прима осовина средњег ваљка је:

$$F_q = 60 \cdot \frac{N_r}{100} \cdot \varphi_D + \frac{G_{ovr}}{3} \qquad (15)$$

У изразу је са N_r обележено оптерећење слога од тежине траке и тежине материјала, са G_{ovr} је обележена тежина обртних делова слога, док ознака φ_D представља динамички фактор удара.

Осим ове силе, треба узети у обзир да осовина додатно прима и инерцијалну силу услед ексцентричности омотача ваљка. При том се сматра да се центар ротације подудара са центром спољашњег круга омотача, док је центар унутрашњег круга омотача померен за вредност ε (слика 9), где ε представља максимално дозвољено одступање дебљине омотача од назначене вредности. Овакав пресек важи по целој дужини ваљка.



Слика 9. *Ексцентричност омотача ваљка*

На основу претходне претпоставке долази се до релације:

$$F_c = 4 \cdot l_{vr} \cdot \rho_{č} \cdot \varepsilon \cdot \frac{(d-2h)^2}{d^2} \cdot v^2 \qquad (16)$$

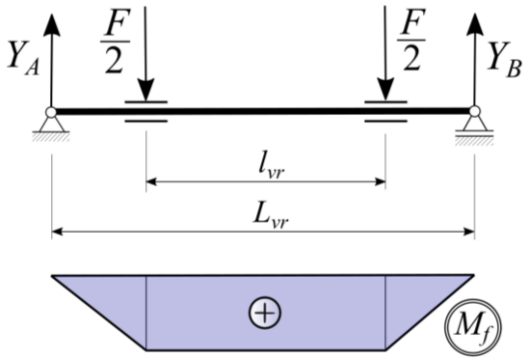
Величине које фигуришу у изразу су: дужина ваљка (l_{vr}), густина материјала омотача ваљка – густина челика ($\rho_č$), максимално дозвољено одступање дебљине зида омотача ваљка од номиналне вредности (ε), пречник спољашњег круга ваљка (d), дебљина зида ваљка (h) и обимна брзина ваљка, тј. брзина траке транспортера (v).

Дакле, укупна сила коју прима осовина средњег ваљка је:

$$F = F_q + F_c \qquad (17)$$

*Напомена: Оптерећења при нетачном монтирању ваљака, као и оптерећења услед нагомилавања материјала на омотаче ваљака имају мали утицај, и у овом прорачуну се занемарују.

Оптерећење се са омотача ваљка преноси на осовину на местима њиховог контакта (у овом случају преко лежајева – слика 10), па као и код провере вратила бубња, сматра се да се оптерећење расподељује на два једнака дела (слика 10).



Слика 10. *Дејство сила на осовину ваљка*

На основу слике 10, јасно се види да је вредност максималног момента једнака:

$$M = \frac{F}{2} \cdot \frac{(L_{vr} - l_{vr})}{2} \qquad (18)$$

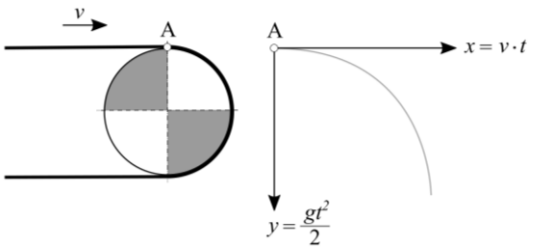
3.5. Истоварни уређај

Истовар материјала се врши преко чеоног бубња. До вредности граничне брзине (она брзина на којој се честице одвајају на наилазном краку траке на бубањ) се долази помоћу образца [1]:

$$v_g = \sqrt{\frac{gD}{2}} \qquad (19)$$

У изразу фигуришу пречник бубња (D) и убрзање земљине теже (g).

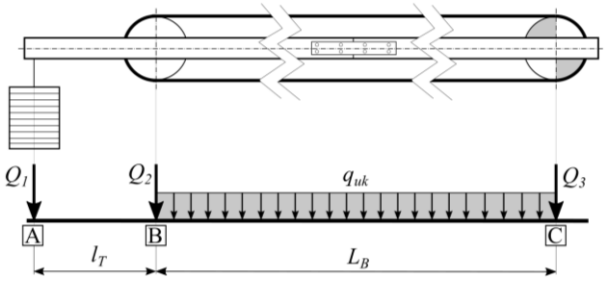
Добија се да је брзина траке већа од граничне, што значи да се материјал одваја у наилазној тачки траке на бубањ, па се даље кретање одвија по закону хоризонталног хица (слика 11).



Слика 11. *Истовар материјала*

3.6. Носећа конструкција

На носач делују континуална оптерећења и концентрисана оптерећења (слика 12).



Слика 12. *Оптерећења носеће конструкције*

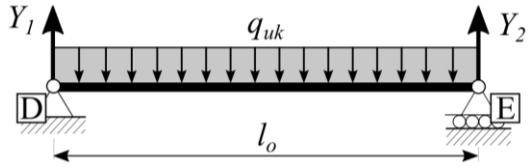
Тежина траке, тежина материјала и тежина самог носача се убрајају у континуална оптерећења. Ради једноставности прорачуна, и тежине ваљчаних слогова ће се убрајати у исту групу. Дакле, укупно континуално оптерећење је:

$$q_{uk} = 2 \cdot q_0 + q + 2 \cdot q_n + q_v \qquad (20)$$

Чланови који фигуришу у изразу су: тежина траке по дужном метру (q_0), тежина материјала по дужном метру (q), тежина носача по дужном метру (q_n) и тежина ваљчаних слогова по дужном метру (q_v).

Што се тиче концентрисаних оптерећења, постоје силе тежина бубњева заједно са вратилом/осовином, тежина тега, као и тежина моторедуктора.

Посматра се део носача од тачке *B* до тачке *C* (слика 12). На овом делу се сви ослонци постављају на растојањима $l_o = 6\ \mathrm{m}$, па се сваки сегмент може посматрати независно, као греда на два ослонца (слика 13).



Слика 13. *Сегмент носача као греда на два ослонца*

На посматрани сегмент делује само једно оптерећење, и то је континуално оптерећење q_{uk}.

4. БЕЗБЕДНОСТ И ЗАШТИТА НА РАДУ

Предвиђене безбедносне мере подразумевају следеће активности:

- извршити прорачун носеће конструкције транспортера;
- извршити проверу погонског механизма као и проверу осталих делова транспортера (траке, бубњеви, ваљци...);
- саставити упутство за употребу и одржавање;
- организовати обуку радника;
- употребити безбедносне уређаје који су предвиђени да сведу ризик од повреда на најмању могућу меру.

Предвиђени безбедносни уређаји на тракастом транспортеру су:

- сигурносни уређај са ужетом;
- тастери за хитно искључење;
- заштитне мреже;
- пролази;
- заштитни елементи на бубњевима и ваљцима.

5. ЗАКЉУЧАК

Рад обухвата комплетан прорачун тракастог транспортера задатих основних параметара. У првом делу наводе се сви битни технички подаци о транспортеру, а даље се врши комплетна провера његових елемената.

6. ЛИТЕРАТУРА

[1] Јован Владић, *Механизација претовара II - Машине и уређаји непрекидног транспорта,* Нови Сад, 1991.
[2] Стандард, *DIN-22101-2011*
[3] *Конвейер ленточный крутонаклонлонный (угол наклона 600)* https://works.doklad.ru/view/Eo3ptPPSdac/9.html
[4] Синиша Кузмановић, *Машински елементи*, Нови Сад, 2012.
[5] Војислав Милтеновић, *Машински елементи,* Ниш, 2009.
[6] Стандард, *is.11592.2000*
[7] Зоран Петковић, Давор Острић, *Металне конструкције у машиноградњи,* Машински факултет, Универзитет у Београду
[8] Марин Антоловић, *Прорачун носиве конструкције*, Факултет стројарства и бродоградње, завршни рад.

Кратка биографија:

Слободан Илкић рођен је у Сомбору 1994. год. Дипломски рад из области машинства одбранио је 2017. год.

контакт: ilkic.slobodan1994@gmail.com

Драган Живанић рођен у Сремској Митровици 1972. год. Докторирао је 2012. год, а од 2019. ради као ванр. професор на Факултету техничких наука у Новом Саду

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.74
DOI: https://doi.org/10.24867/07AM05Trifunovic

UTICAJ OBLIKA RAZVODNIKA NA BRZINU LIVA PRI GRAVITACIONOM LIVENJU ALUMINIJUMA

THE INFLUENCE OF THE RUNNER SHAPE ON MELT VELOCITY IN ALUMINUM GRAVITY CASTING

Slaviša Trifunović, Lazar Kovačević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – PROIZVODNO MAŠINSTVO

Kratak sadržaj – *Kvalitet aluminijumskih odlivaka u velikoj meri zavisi od oblika ulivnog sistema koji treba da obezbedi kontrolisano ulivanje brzinom nižom od kritične. U radu su prezentovani rezultati merenja brzine rastopljenog aluminijuma na izlazu iz razvodnika pomoću metode praćenja trajektorije pri slobodnom isticanju iz horizontalnog kanala. Istraživanja su sprovedena za različite oblike razvodnika.*

Ključne reči: *Popunjavanje kalupa, Ulivni sistem, brzina rastopa*

Abstract – *Quality of aluminum alloy castings rely on the gating system design and its ability to provide controlled filling with lower than critical velocities. Runner exit velocities for several runner shapes were calculated by the trajectory tracking method by ejecting the melt through the open horizontal runner.*

Keywords: *Mold filling, gating system, metal velocity*

1. UVOD

Prilikom ulivanja rastopljenog metala dolazi do zahvatanja gasova i oksida, nemetalnih uključaka i drugih nečistoća. Iz tog razloga može se smatrati de je većina rastopljenih metala puna unutrašnjih grešaka koje značajno smanjuju njegove mehaničke osobine [1]. Iako veliki uticaj na kvalitet odlivka ima početno stanje rastopa u peći, odnosno ulivnoj kašici, površinska turbulencija unutar ulivnog sistema takođe ima značajnu ulogu. Da bi se smanjio procenat grešaka i dobio kvalitetan odlivak neophodno je projektovati optimalan ulivni sistem koji ima za zadatak obezbeđivanje prenosa čistog rastopa iz lonca u kalupnu šupljinu, a da nakon toga ne dođe do reoksidacije ili povećavanja sadržaja gasova. Osnovni zadaci ulivnog sistema su [2,3]:

- kontrola brzine ulivanja u kalupnu šupljinu kako ne bi došlo do hladnih zavara i nedolivenosti u odlivku
- popunjavanje kalupne šupljine tako da se ne pojavi zahvatanje gasova, šljake i oksida i njihov ulazak u kalupnu šupljinu;
- sprečavanje erozije kalupa;
- minimalan utrošak materijala;
- lako odvajanje od odlivka pri čišćenju;
- rastopljen materijal je potrebno uliti u kalupnu šupljinu tako da obezbedi usmereno očvršćavanje odlivka ka hraniocu. Rastop koji poslednji ulazi u kalup ima najveću temperaturu te je iz tog razloga neophodno pozicionirati ulivni sistem na mestu gde se očekuje zadnje očvršćavanje odlivka.
- sprečavanje deformacija odlivka. Ovaj problem je posebno izražen kod tankozidih odlivaka gde nepravilan raspored toplote dovodi do nepoželjnog načina očvršćavanja i vitoperenja odlivka, dok skupljanje ulivnika tokom hlađenja može prouzrokovati pojavu vrućih prslina u odlivku.

Tokom vremena, razvijen je veliki broj oblika ulivnih sistema koji nude mogućnost ispunjavanja navedenih zadataka. Svaki od njih karakterišu četiri osnovna elementa [3]:

- ulivna čaša,
- sprovodnik,
- razvodnik ,
- ulivnik, ulivni kanal.

Pored njih, u cilju povećanja kvaliteta konačnog odlivka, mogu se projektovati i neki dodatni elementi kao što su prigušnice, kolektori, filteri, hvatači šljake, produžeci razvodnika i sl. Konačan oblik ulivnog sistema zavisi od više faktora, od kojih su za najznačajniji [4]:

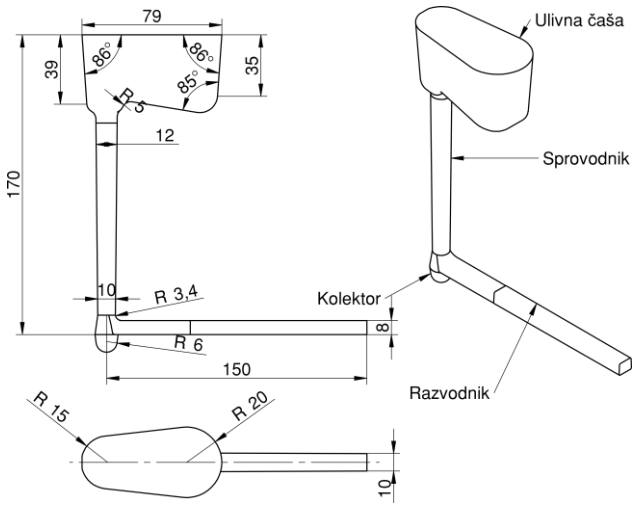
- primenjena tehnologija (postupak) livenja,
- geometrija odlivka,
- vrsta materijala.

Novija istraživanja pokazuju da je za dobijanje dobrih mehaničkih osobina aluminijumskih odlivaka, neophodno smanjiti brzinu rastopa u ulivnom kanalu na oko 0,5 m/s [5]. Ovaj uslov je veoma teško zadovoljiti jer se navedena kritična brzina dostiže već nakon pada rastopa sa visine od oko 13 mm [1]. Skoro svi odlivci koji se industrijski proizvode su značajno većih dimenzija i zahtevaju upotrebu sprovodnika koji su viši za barem jedan red veličina, a u velikom broju slučajeva i za dva reda veličina. Stoga se pri projektovanju ulivnog sistema velika pažnja mora posvetiti načinima za usporavanje liva pre izlaska iz ulivnog kanala. Kako bi se ulivni sistem mogao pravilno dimenzionisati tako da omogući ispunjavanje navedenog uslova, neophodno je što preciznije poznavanje gubitaka koji se javljaju u različitim elementima ulivnog sistema. Predstavljeni rezultati predstavljaju deo šireg istraživanja sa ciljem da se oni precizno definišu.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio doc. dr Lazar Kovačević

2. METODOLOGIJA

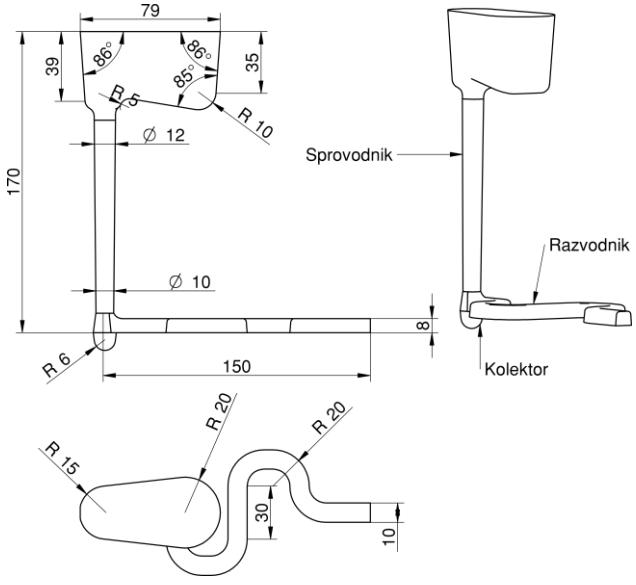
Osnovni ispitivani ulivni sistem je prikazan na slici 1. Proračun je izveden uzimajući u obzir trenutno dostupne tablice koje za jednostavne ulivne sisteme definišu koeficijent isticanja od 0,8 [6].



Slika 1. *Izgled osnovnog ulivnog sistema (ravan normalni)*

Zadržavajući isti oblik ulivne čaše, sprovodnika i osnove sprovodnika, brzina isticanja rastopa je merena za još tri ulivna sistema koji se od osnovnog (nazvanog ravan normalni) razlikuju u sledećem:

- razvodnik je sa 8 mm povišen na 16 mm – u daljem tekstu nazvan ravan povišen
- razvodnik je urađen sa dodanim krivinama kako bi se usporio rastop – u dalje tekstu nazvan krivudavi normalan, slika 2.
- razvodnik je urađen sa dodatnim krivinama kao i krivudavi normalan, ali uz dodatno povišenje razvodnika sa 8 mm na 16 mm.



Slika 2. *Izgled ulivnog sistema sa dodatim krivinama u razvodniku (krivudavi normalni)*

Za potrebe kalupovanja segmenti modela ulivnog sistema su izrađeni deponovanjem istopljenog filamenta. Ulivni sistem je izrađen modularno kako bi mogao da se nesmetano vadi iz kalupa, a pojedini elementi koriste više puta u različitim varijacijama. Kalupnice u kojima su izrađivani kalupi su napravljene sa šarkama tako da se mogu otvoriti i osloboditi kalup. Na taj način omogućeno je nesmetano isticanje liva iz kalupa.

Klupi su izrađeni od smeše gipsa, peska i vode u zapreminskom odnosu $7:7:4{,}5$ čime je omogućeno višestruko ulivanje u isti kalup. Dno kalupa u koji se ulivao rastop se nalazilo na visini od 317 mm od poda. Pomoću termopara koji je postavljen u ulivnu čašu praćena je temperatura rastopa kako bi se isključio uticaj promene temperature na viskoznost liva. Zidovi kalupnih šupljina su postavljani paralelno u odnosu na papir na kome je odštampana mreža dimenzija 10 x 10 mm. Podeona ravan kalupa postavljana je u horizontalan položaj pomoću libele, slika 3.

Slika 3. *Dovođenje podeone ravni u horizontalni položaj*

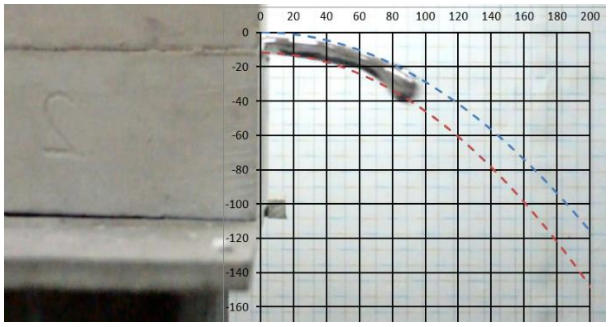
Svako ulivanje je snimano kamerom postavljenom upravno u odnosu na pravac isticanja. Objektiv kamere je podešavan tako da se njegova osa poklapa sa mestom isticanja rastopa. Snimci isticanja su obrađivani u programu „Media player classic home cinema“ gde su isecane slike za potrebe određivanja brzina isticanja u pravilnim vremenskim intervalima u odnosu na prvi frejm na kojem se primećuje liv. Ovako dobijene slike su naknadno obrađivane u programskom paketu „GNU-Image Manipulation Program“ gde je vršena njihova rotacija kako bi se uklonila eventualna greška položaja objektiva kamere. Obrađene slike su unošene u program „Microsoft Excel″ u kojima je vršena procena brzine isticanja, slika 3. Procena je vršena poređenjem trajektorije tečnog rastopa sa idealnom trajektorijom horizontalnog hica koja je dobijena pomoću izraza (1):

$$y_i = -\frac{g \cdot x_i^2}{2v^2}, \quad i = 1 \ldots 240 \qquad (1)$$

gde su: x_i i y_i dekartove koordinate posmatrane tačke trajektorije ($x_i = 0{,}001 \cdot i\,[m]$), v - početna brzina isticanja liva, g – gravitaciono ubrzanje.

Vrednost početne brzine isticanja je varirana sve dok se nije postiglo poklapanje teorijske trajektorije horizontalnog hica sa oblikom struje tečnog metala na slici 4. Minimalna i maksimalna brzina isticanja određivane su iz razloga što je otežano pronalaženje linije srednje brzine, te je ovom metodom pouzdanije iznalaženje srednje brzine isticanja. Ova procedura je ponavljana za svaki razmatrani vremenski trenutak i za svako ulivanje.

Kako bi se imao uvid u ponovljivost rezultata planirano je vršenje tri ponavljanja za svaku varijaciju oblika razvodnika.



Slika 4. *Primer određivanja maksimalne i minimalne brzine isticanja u programu Microsoft Excel*

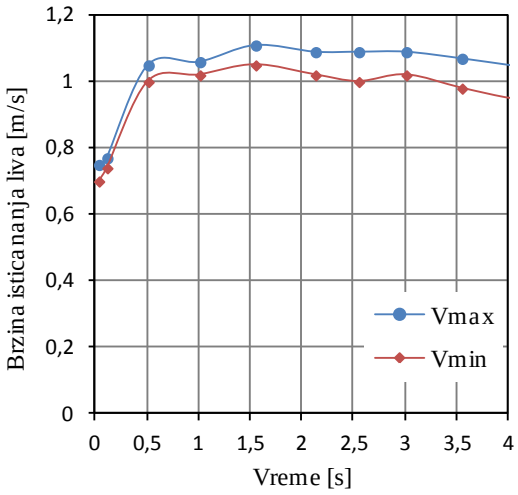
3. REZULTATI I DISKUSIJA

Termoparovi postavljeni u ulivnu čašu pokazali su zadovoljavajuću ponovljivost temperature ulivanja. Prosečna temperatura ulivanja je iznosila 672±15°C. Prilikom izvođenja eksperimenta, usled aksijalnog skupljanja odlivka dobijenih krivudavim razvodnicima dolazilo je do oštećenja kalupa. Na temperaturama ulivanja aluminijuma dolazi do slabljenja gipsanog veziva i prilikom vađenja odlivaka iz kalupa dolazilo je do njegovog oštećenja, zbog čega je postajao neupotrebljiv za dalje eksperimente. Broj izvedenih ulivanja za različite oblike razvodnika dat je u tabeli 1.

Tabela 1. *Izvedeni broj ulivanja za svaki oblik razvodnika*

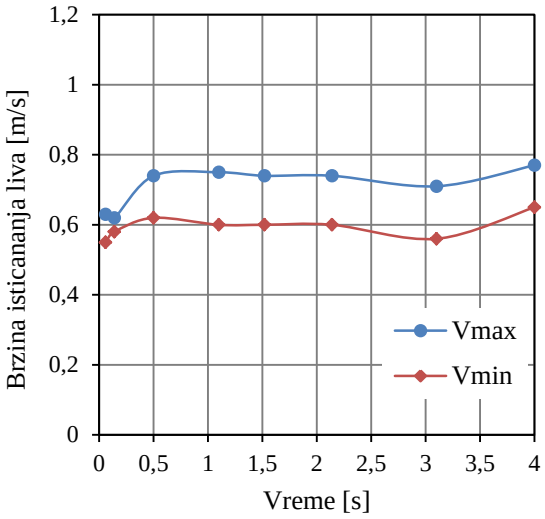
Oblik razvodnika	Broj ulivanja
Ravan normalan (RN)	3
Ravan povišen (RP)	3
Krivudav normalan (KN)	2
Krivudav povišen (KP)	1

Tipičan oblik promene brzine isticanja liva iz kalupa prikazan je na slikama 5 i 6.



Slika 5. *Primer dijagrama brzine isticanja za ravan normalan razvodnik*

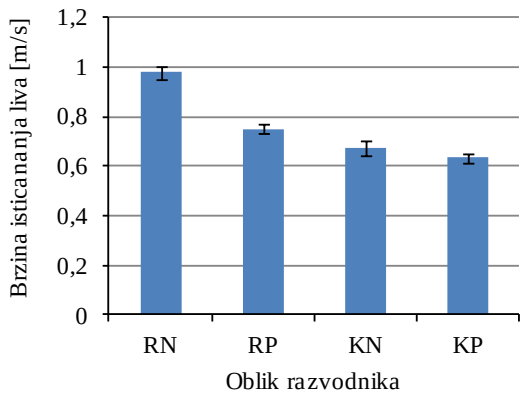
U prvih pola sekunde brzina liva raste dok se ne uspostavi regularan režim strujanja metala. Ovaj rast brzine je najverovatnije povezan sa procesom popunjavanja ulivne čaše i potpunim ispunjavanjem ulivnog sistema tečnim livom. Nakon tog početnog vremena, procenjene brzine isticanja su relativno ujednačene, ali ne i potpuno konstantne. Pretpostavlja se da je ovo oscilovanje posledica blage varijacije u visini liva u ulivnoj čaši koje se pod dejstvom gravitacije pretvara u varijaciju brzine isticanja. Iako se pri ulivanju rastopa vodilo računa o ovoj pojavi, činjenica da je ulivanje vršeno ručno onemogućila je apsolutnu kontrolu visine. Za dobijanje preciznijih rezultata merenja u budućim istraživanjima moguće je izmeniti postojeću geometriju ulivne čaše dodavanjem prelivnog kanala, poput one korišćene u referenci [7].



Slika 6. *Primer dijagrama brzine isticanja za krivudav normalan razvodnik*

Analizom dijagrama prikazanih na slikama 5 i 6 uočljivo je da se za istu veličinu poprečnog preseka brzine isticanja liva smanjuju dodavanjem krivina u razvodnik. Ovakav rezultat i jeste bio očekivan s obzirom da svaka promena pravca tečenja struje rastopa dovodi do određenih energetskih gubitaka. Međutim, postavlja se pitanje da li su navedene razlike statistički značajne i da li je razvijena eksperimentalna postavka dovoljna da ovu razliku i dokaže. Stoga su za svako ulivanje za određeni oblik hranioca objedinjene sve izmerene brzine, (uključujući maksimalne i minimalne) i za njih je određena srednja vrednost, standardna devijacija i 95% interval poverenja srednje vrednosti. Kako bi se izbegla greška početnog popunjavanja ulivnog sistema, u razmatranje su uzete samo brzine nakon 0,5s od početka isticanja. Dobijeni rezultati prikazani su na slici 7. Može se uočiti da je interval poverenja relativno uzak uzimajući u obzir razlike u izmerenim brzinama isticanja. Stoga se može konstatovati da je razvijena eksperimentalna metodologija adekvatna za datu namenu.

Analizom rezultata prikazanih na slici 7 može se uočiti da su najviše brzine isticanja povezane sa ravnim normalnim razvodnikom, dok je najniža brzina isticanja ostvarena primenom krivudavog povišenog razvodnika. Krivudavi razvodnici su bili u stanju da snize brzine strujanja liva u znatno većoj meri nego što su to ostvarili ravni razvodnici. Prosečna brzina isticanja iz ravnog normalnog razvodnika iznosila je 0,98 m/s, što je niže od očekivane brzine koja za koeficijent isticanja od 0,8 iznosi 1,38 m/s. Istovremeno, prosečna brzina isticanja iz ravnog povišenog razvodnika od 0,75 m/s ukazuje da se u tom slučaju koeficijent isticanja može proceniti na oko 0,55.



Slika 7. *Srednje vrednosti brzine isticanja za svaki oblik razvodnika. Trake sa greškama pokazuju 95% interval poverenja*

Dobijena razlika se može objasniti na sledeći način. Realni koeficijent isticanja koji ukazuje na gubitke ravnog ulivnog sistema iznosi oko 0,55. Međutim, kako bi ulivni sistem bio u stanju da iskoristi sve prisutne energetske gubitke, neophodno je da nakon svakog usporavanja liva dođe do proširenja poprečnog preseka posmatranog elementa. U suprotnom, ulivni sistem ima ulogu mlaznice koja efektivno povećava brzinu strujanja liva [1,5,7]. Ravni povišen ulivni sistem je bio predimenzionisan, nije bilo efekta mlaznice i svi energetski gubici unutar ulivnog sistema su doveli do efikasnog smanjenja brzine strujanja liva.

Slika 8. *Isticanje rastopa iz ravnog povišenog razvodnika*

Analizom snimaka struje tečnog metala snimljenih u pravcu isticanja, uočeno je da povišeni razvodnici na izlazu nisu bili potpuno ispunjeni tečnim metalom, slika 8. Ovo je dovelo zahvatanja vazduha u razvodniku i povremenog prekida struje metala, čime je precizno očitavanje brzina strujanja bilo otežano. Za očekivati je da bi ovako dobijena struja tečnog metala izmešana sa vazduhom dovela do nižih mehaničkih osobina odlivka. Stoga se ne preporučuje upotreba povišenih razvodnika, već je neophodno proračunati tačnu geometriju istih, uzimajući u obzir eksperimentalno utvrđen koeficijent isticanja od 0,55. Treba napomenuti da je sličan problem sa zahvatanjem vazduha primećen i kod krivudavog normalnog razvodnika koji je na izlazu bio potpuno ispunjen livom. Pretpostavlja se da je uzrok ovoj pojavi loš spoj na podeonoj ravni i mogućnost zahvatanja vazduha u zonama potpritiska u unutrašnjoj zoni krivine.

4. ZAKLJUČCI

[1] Razvijena eksperimentalna postavka je adekvatna i može se koristiti za određivanje brzina strujanja rastopa na izlasku iz razvodnika ili ulivnih kanala.
[2] Kako bi se smanjio uticaj varijacije nivoa liva u ulivnoj čaši, a u cilju dodatnog povećavanja preciznosti merenja, u budućim istraživanjima se preporučuje primena ulivne čaše sa prelivnim kanalom.
[3] Dobijeni rezultati pokazuju da je koeficijent isticanja liva kod manjih i jednostavnih ulivnih sistema niži od trenutno preporučenih 0,8. Primena navedenog koeficijenta isticanja pri proračunu ulivnog sistema dovodi do nedovoljnog iskorišćavanja svih energetskih gubitaka prisutnih u ulivnom sistemu.
[4] Dodavanjem krivina u okviru razvodnika moguće je postići dodatno usporavanje liva. Međutim, u okviru ovog istraživanja primena krivudavih razvodnika izazvala je povećano zahvatanje gasova.

5. LITERATURA

[1] J. Campbell, " *Complete Casting Handbook: Metal Casting Processes, Metallurgy, Techniques and Design"*, Oxford, Elsevier Butterworth-Heinemann, 2015.
[2] L. Kovačević, "*Savremene tehnologije livenja - skripta*", Novi Sad, Fakultet tehničkih nauka, 2018.
[3] R. Kovač, "*Tehnologija izrade odlivaka*", Novi Sad, Fakultet tehničkih nauka, 2006.
[4] P. Beeley, "*Foundry technology*", Oxford, Elsevier Butterworth-Heinemann, 2001.
[5] J. Campbell, "*Castings Practice: The Ten Rules of Castings"*, Oxford, Elsevier Butterworth-Heinemann, 2004.
[6] I. Galić, I. Katavić, I. Kerekeš, J. Pirš, *"Ljevački priručnik"*, Zagreb, Savez ljevača Hrvatske, 1985.
[7] F.-Y. Hsu, M. R. Jolly, J. Campbell, " A multiple-gate runner system for gravity casting", *Journal of Materials Processing Technology*, Vol. 209, pp. 5736-2750, 2009.

Kratka biografija:

Slaviša Trifunović rođen je u Loznici 1995. godine. Diplomirao je 2018. godine na Fakultetu tehničkih nauka u Novom Sadu iz oblasti mašinskog inženjerstva - studijski program Proizvodno mašinstvo. kontakt: zeko.trifunovic@gmail.com

Lazar Kovačević rođen je u Beogradu 1981. godine. Doktorirao je na Fakultetu tehničkih nauka 2015. god., gde od 2016. godine radi u zvanju docenta.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.7/.9
DOI: https://doi.org/10.24867/07AM06Babic

KONFIGURISANJE UPRAVLJAČKOG SISTEMA NUMERIČKI UPRAVLJANE MAŠINE ALATKE SA HIBRIDNOM KINEMATIKOM

CONFIGURING THE CONTROL SYSTEM OF NUMERICALLY CONTROLLED TOOLS MACHINE WITH HYBRID KINEMATICS

Aleksandar Babić, *Fakultet tehničkih nauka, Novi Sad*

Oblast: PROIZVODNO MAŠINSTVO

Kratak sadržaj – *Konfiguracija O-X glide hibridnog mehanizma, originalnog, mehanizma razvijenog na FTN-u za potrebe gradnje mašina alatki i manipulacionih sistema, podrazumjeva implementaciju nelinearnog kinematskog lanca u upravljački sistem. Hibridni mehanizmi, u koje ovaj mehanizam spada, predstavljaju kombinaciju serijskih i paralelnih mehanizmam formiranu sa ciljem ostvarivanja što boljeg iskorištenja mehaničkih karakteristika. Odabrani upravljački softver, LinuxCNC, je otvorene arhitekture, što znači da se mogu izvršiti promjene u jezgru, u zavisnosti od potrebe, da bi se definisala konfiguracija i na taj način primenom besplatnog softvera realizovati ekpserimentalni prototip mašine alatke.*

Ključne reči: *O-X glide, LinuxCNC*

Abstract – *The configuration of the O-X glide hybrid mechanism developed on FTN for the purpose of building machine tools and manipulation systems, involves the implementation of a nonlinear kinematic chain into the control system. The hybrid mechanisms, to which this mechanism belongs, are a combination of serial and parallel mechanisms formed with the aim of maximizing the use of mechanical characteristics. The selected control software, LinuxCNC, has open architecture, which means that changes into core can be made., as needed, to define the configuration and on this way, by implementing free software, to realize experimental prototypes of machine tools.*

Keywords: *O-X glide, LinuxCNC*

1. UVOD

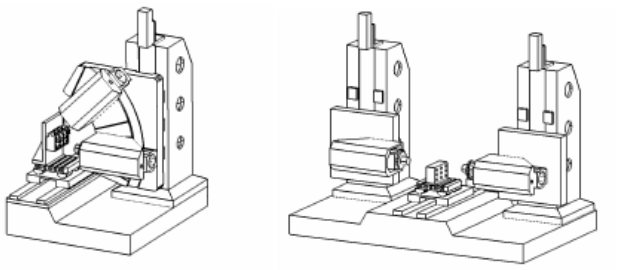
Mašina alatka kao rezultat čovjekove vještine, znanja i svijesti da to može i bolje, razvija se kao posljedica razvoja i usavršavanja proizvodnih tehnologija [2].

Napredak mašina alatki obuhvata povećanje režima obrade kod obrade savremenih materijala, poboljšanje mehaničke strukture mašina, modularnost upravljačkih sistema kao i povezivanje u industrijski koncept Internet stvari (*Industry Internet of things*) [4].

Razvoj mehatroničkih sistema, da bi se dobile što bolje kinematske karakteristike, utiče na stvaranje paralelne kinematike pogodne za implementaciju u mašine alatke, na slici 1. je prikazana jedna takva mašina.

Zatim, napredovanjem modularne strukture mašina alatki nastala je koncepcija mašina alatki koja omogućava stvaranje fleksibilnih struktura koje su prilagodljive potrebama proizvodnje i geometriji samog proizvoda.

Slika 1. *LOLA pn101_4V.1 [4]*



Slika 2. *Rekonfigurabilne mašine alatke [4]*

Hibridni mehanizmi su nastali kao kombinacija serijskih i paralelnih mehanizama. Cilj njihovog razvoja jeste stvaranje kompromisa između serijske i paralelne kinematike. To podrazumjeva maksimalno iskorištenje prednosti ovih mehanizama, uz minimalne nedostatke [3].

Upravljanje i programiranje mašina alatki po tradiciji komjuterskog numeričkog upravljanja se danas može smatrati kao standardizovanom vještinom koju pokriva veći broj specijalizovanih proizvođača komponenti [2].

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

Za to su zaduženi upravljački sistemi ili upravljačke jednice koji predstavljaju jedan od najvažnijih podsistema numerički upravljanih mašina alatki, pa je njihov zadatak izvršavanje određenih funkcija. Upravljački sistemi mogu biti u vidu hardverski realizovanih kontrolnih jedinica (Fanuc, Siemens) kao i u vidu specijalizovanog softvera (LinuxCNC).

O-X glide mehanizam priprada grupi mehanizama sa hibridnom seijsko paralelnom kinematikom. Razvijen je na Fakultetu tehničkih nauka u Novom Sadu [3]. Njegova konfiguracija podrazumjeva glavni osvrt u ovom radu i vrši se u LinuxCNC okruženju.

LinuxCNC je izabran zbog toga što je besplatan softver, otvorene je arhitekture i može da odgovori na sve postavljenje zahtjeve u ovom slučaju, da bi se riješilo pitanje upravljanja O-X glide hibridnog mehanizma

2. O-X GLIDE

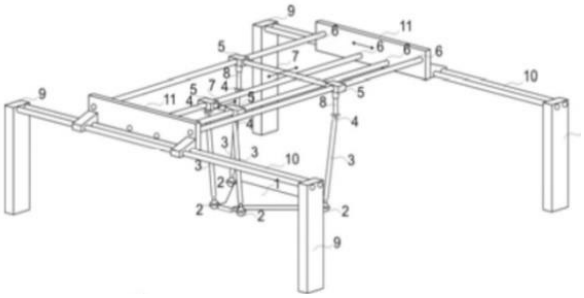
O-X glide hibridni mehanizam čini ravanski paralelni mehanizam čija se baza sastoji od pokretne platforme, koja je preko sfernih zglobova vezana za četiri štapa konstantne dužine.

Na drugom kraju, štapovi su vezani za klizače preko zglobova sa jednim rotacionim stepenom slobode kretanja, svaki klizač se kreće po sopstvenoj vođici. Rapored klizača je parni pa su spojeni krutom vezom sa ciljem dobijanja iste brzine i ubrzanja. Grupe klizača raspoređeni su na različitim rastojanjima u pravcu verikale ose od pokretne platforme, što omogućava njihovo mimoilaženje u ravni, a time i dualnost kretanja mehanizma, odnosno kretanje u opruženom (O) i ukrštenom (X) položaju.

Uvođenjem vertikalnih kompenzacionih elemenata, na višim klizačima, vrši se kompenzovanje na rastojanjima između klizača i pokretne platforme. Tako je ostvareno da, zglobovi između štapova i klizača budu pozicionirani u istoj ravni koja je paralelna sa pokretnom platformom i omogućena je primjena štapova iste dužine što velikim dijelom pojednostavljuje konstrukciju mehanizma.

Na slici 3. prikazan je O-X glide hibrdni mehanizam kao i numerički označeni dijelovi:

1. pokretna platforma;
2. sferni zglobovi;
3. štapovi konstantne dužine;
4. zglobovi sa jednim rotacionim stepenom slobode kretanja;
5. klizači kojima se ostvaruje ravansko kretanje pokretne platforme;
6. vođice klizača kojima se ostvaruje ravansko kretanje pokretne platforme;
7. kruta veza koja spaja klizače (5);
8. kompenzacioni elementi;
9. noseći kontruktivni elementi mehanizma, četiri noge;
10. vođice klizača (11);
11. klizači koji pomjeraju ravanski mehnizam da bi se ostvarilo kretanje duž treće ose, translatorne ose [3].



Slika 3. *Šematski prikaz O-X glide hibridnog mehanizma [3]*

2.1. Kinematika O-X glide hibridnog mehanizma

Tokom dizajniranja mašine alatke neophodno je izvršiti analizu kinematskih faktora prilikom kretanja mehanizma.

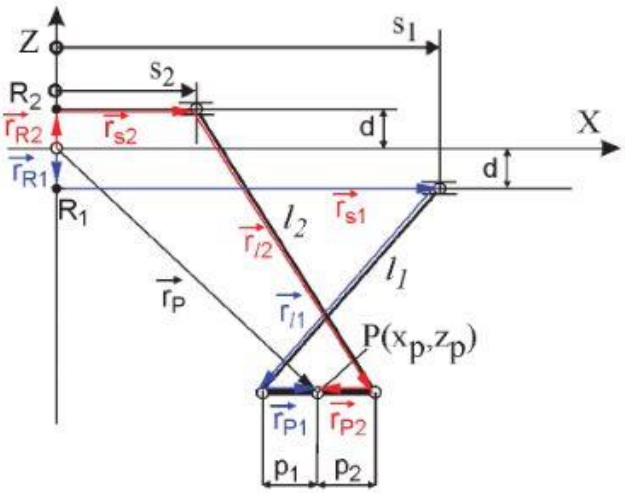
U ovom slučaju se podrazumjeva analiza brzina pojedinih elemenata mehanizma za maksimalne vrijednosti koje postiže pokretna platforma u mašini alatki [5].

Da bi se sproveo adekvatan izbor vođica, klizača i spojeva, elementa napajanja mašine i dr., neophodna je analiza ubrzanja koja se javljaju na elementima u eksplatacionim uslovima, ali pod maksimalnim opterećenjem pokretne platforme.

Kako bi se izvršila kinematska analiza mehanizma klizanja O-X glide hibridnog mehanizma koristi se inverzna kinematika, koja podrazumjeva podešavanje kretanja pokretne platforme kao i praćenje ponašanja svih zajedničkih elemenata mehanizma. Ovako se mogu dobiti brzine i ubrzanja poprečnih klizača za kretanje pokretne platforme preko teoretskog radnog prostora definisanog brzinom i ubrzanjem.

Kinematska analiza O-X glide hibridnog mehanizma podrazumjeva rješavanje njegove inverzne i direktne kinematike u oba njegova oblika.

Na slici 4. prikazan je kinematički model O-X glide hibridnog mehanizma na osnovu koga je izvršen proračun kako bi se riješila inverzna i direktna kinematska problematika [5].



Slika 4. *Kinematički model O-X glide hibridnog mehanizma [5]*

Inverzna kinematika [5]:

- Za ukršteni oblik O-X glide hibridnog mehanizma:

$$s_1 = x_p - p_1 + \sqrt{l_1^2 - (z_p + d)^2} \quad (1)$$

$$s_2 = x_p + p_2 - \sqrt{l_2^2 + (z_p - d)^2} \quad (2)$$

- Za opuženi oblik O-X glide hibridnog mehanizma:

$$s_1 = x_p - p_1 - \sqrt{l_1^2 - (z_p + d)^2} \quad (3)$$

$$s_2 = x_p + p_1 + \sqrt{l_2^2 - (z_p - d)^2} \quad (4)$$

Direktna kinematika [5]:

$$z_p = \frac{-m_6 - \sqrt{m_6^2 - 4m_5 m_7}}{2m_5} \quad (5)$$

$$x_p = m_3 + m_4 * z_p \quad (6)$$

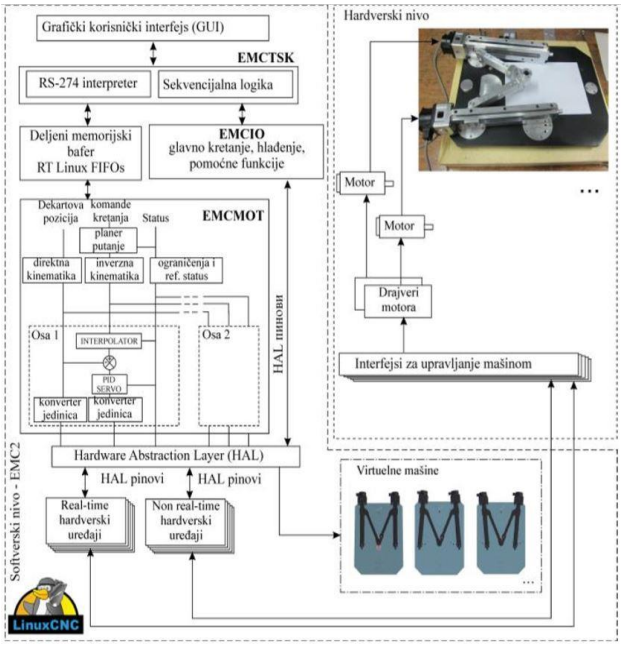
3. LINUXCNC

Programski sistem LinuxCNC (stariji naziv EMC, (*Enhanced machine Controller* - Unaprijeđeni mašinski kontroler), ima mogućnost upravljanja mašinama alatkama i robotima kroz kontrolu step i servo motora i drugih uređaja vezanih za kontrolu kretanja.

LinuxCNC može da kontroliše do devet numerički upravljanih osa u koordinatnom kretanju [7].

Programski sistem LinuxCNC sadrži pet osnovnih komponenti:

- (EMCMOT) kontroler pokreta;
- (EMCIO) diskretni I/0 regulator;
- (EMCTASK) izvršilac zadataka koji ih koordinira;
- (GUI) grafičko korisnički interfejs;
- (HAL) hardverski sloj apstrakcije [1].

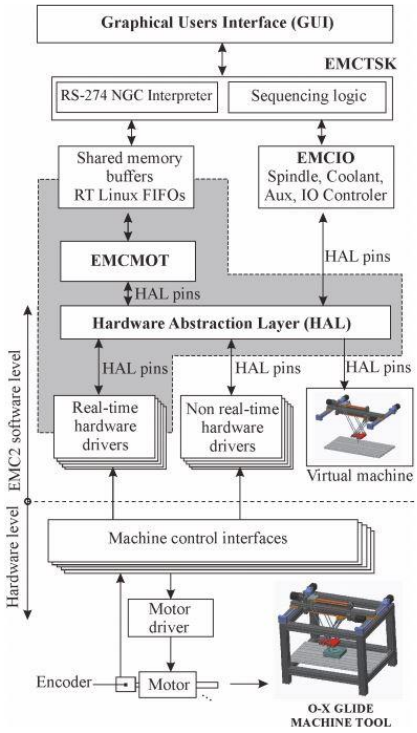


Slika 5. *Struktura CNC sistema, otvorena arhitekture na bazi LinuxCNC softvera [1]*

4. KONFIGURACIJA O-X GLIDE HIBRIDNOG MEHANIZMA

Formiranje konfiguracije je mali istraživački zadatak za svaku mašinu alatku i najčešći je slučaj kada je u pitanju mali budžet projekta u lokalnim uslovima, pa se sprovodi korištenjem sopstvenih resursa [5].

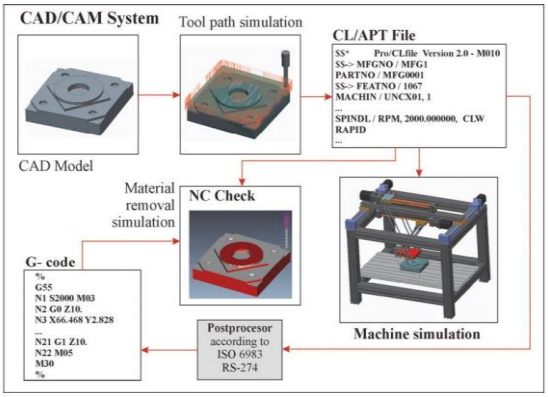
Da bi se mašine sa hibridnom kinematikom kontrolisale neophodno je primjeniti inverzna i direktna rješenja za kontrolu kinematike, pa je pogodnost izabranog LinuxCNC softvera modifikacija i distribucija koda.



Slika 6. *Struktura programskog sistema zasnovanog na CAD/CAM [5]*

Promjene, odnosno modifikacije u jezgru podrazumjevaju zamjenu uobičajenih standardnih trivijalnih funkcija inverznih i direktnih kinematika, odgovarajućim funkcijama. Programiranje navedenih funkcija vrši se pomoću programskog jezika C, upotrebljavajući odgovarajuću koriničku datoteku za kinematiku i uključivanje ove datoteke u LinuxCNC-u, u modulu EMCMOT [5].

Integrisanjem modela upravljanja, parametri mašine su takođe definisani, kao i referentni položaj svih osa, a zatim realizacija kompajliranja i povezivanja softvera.



Slika 7. *Struktura programskog sistema zasnovanog na CAD/CAM [5]*

Svaka mašina opisana je u LinuxCNC sa nekoliko konfiguracijskih datoteka. HAL datoteka (.hal) sadrži potpune informacije o načinu kao na koji LinuxCNC djeluje sa povezanim hardverom. Među tim hardverskimm komponentama spadaju [6]:

- akutatori;
- vretena;
- referentna tačka;
- senzori;
- ulazi i izlazi itd [6].

Što se tiče konfiguracionih datoteka mašine (.ini) one opisuju parametre mašine:

- koordinate;
- ograničenja spojeva;
- poziciju referentne tačke
- bzine i ubrzanje osa;
- PID pojačavanje
- mjerni sistem itd [6].

Kada su u pitanje mašina sa netrivijalnom kinematikom potrebno je obezbjediti (.c) datoteku koja izvodi inverzne i direktne transformacije koordinata.

4. ZAKLJUČAK

Kao značajna rješenja ovog master rada smatra se: opisana izvodljivost tehnologije virtuelnog prototipa, primjena O-X glide hibridnog mehanizma, smanjenje troškova uz korištenje LinuxCNC softvera otvorene arhitekture i sl. Radom je obuhvaćen razvoj O-X glide mehanizma od skice do definisanja svih statičkih i kinematskih aktivnih komponenata, na osnovu čega se vrše detaljnije analize radnog prostora kao i kinematske karakteristike pokretnih elemenata zasnovanih na direktnoj i inverznoj kinematici. Na osnovu istraživanja moguće je zaključiti da se O-X glide hibridni mehanizam može upotrebljavati pri obradi dijelova prizmatičnog oblika manje visine.

LinuxCNC kao izabrani upravljački sistem, podrazumjeva koncept PC – CNC upravljački sistem, odnosno operativni sistem u realnom vremenu, pa kao takav ima izuzetan potencijal u razvoju mašina alatki složene kinematske strukture u sistemima koji zahtjevaju veću fleksibilnost i rekonfiguraciju, što je potvrđeno u ovom slučaju.

5. LITERATURA

[1] S. Živanović, Z. Dimić, G. Vasilić, B. Kokotović, Konfigurisanje virtuelne rekonfigurabilne dvoosne mašine sa paralelnom kinematikom inteegrisane sa CNC sistemom otvorene arhitekture na bazi EMC2 softvera, *TEHNIKA – MAŠINSTVO 67 (2018) 4*, Beograd, 2018.

[2] S. Živanović, Konfigurisanje novih mašina, doktorska disertacija, mašinski fakultet univerziteta u Beogradu, Beograd, 2010.

[3] C. Mlađenović, S. Tabaković, M. Zeljković, Radni prostor 0-X glide hibridnog mehanizma, DOI: 10.13140/2.1.2874.7522 INFOTEH-JAHORINA, Jahorina, BiH, Vol.: 12, March 2013

[4] S. Tabaković, M. Zeljković, S. Živanović, Savremene mašine alatke – trendovi u edukaciji, Primena novih tehnologija i ideja u školskom inženjerskom obrazovanju, Požega, Srbija, May 15-16, 2017.

[5] S. Živanović, S. Tabaković, M. Zeljković, Configuring a machine tool based on hybrid o-x glide mechanism, *Machine design*, vol.8(2016) no.4, issn 1821-1259 pp. 141-148, January 2016.

[6] A. Kyrychenko, Open source CNC control for parallel kinematic machine tool, Техніка в сільськогосподарському виробництві, галузеве машинобудування, автоматизація*, вип. 26, 2013р,* УДК 62-231:621.9.04, Kirovograd National Technical University.

Kratka biografija:

Aleksandar Babić rođen je u Trebinju 1996. god. Master rad na Fakultetu tehničkih nauka iz oblasti Proizvodnog mašinstva – Računarom podržane tehnologije odbranio je 2019.god.

kontakt: aleksandarbabic03@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 628.8
DOI: https://doi.org/10.24867/07AM07Bartula

VRELOVODNI SISTEM DALJINSKOG GRIJANJA

DISTRICT HEATING HIGH TEMPERATURE SISTEM

Nikola Bartula, *Fakultet tehničkih nauka, Novi Sad*

Oblast - MAŠINSTVO

Kratak sadržaj – *U radu su date sve vrste pojedinačnih i centralnih načina grijanja, s tim da je akcenat stavljen na daljinsko vrelovodno grijanje. Prikazani su elementi daljinskog sistema, njegov način funkcionisanja, namena, koristi i mane. Pored teoretskog dijela, rad u sebi sadrži primjer projektovanja vrelovodne trase daljinskog grijanja, koja se pruža Dunavskom i ulicom Ive Lole Ribara, sa svim potrebnim proračunima neophodnim za izradu idejnog projekta.*

Abstract - *In this work are given all types of individual and central heating sistems, with the accent placed on district heating. Elements of the district system, its mode of operation and purpose are shown in this work, with all advantages and disadvantages. In addition to the theoretical part, the work contains an example of designing a hot-water district heating line, which is placed on the Duavska and Ive Lola Ribara streets, with all the necessary calculations for the conceptual design.*

Keywords: *Heating, District heating sistems, designing*

1. UVOD

Pojam grijanja podrazumijeva obezbjeđivanje adekvatnih uslova u prostoriji u kojoj borave ljudi, kako bi boravak bio što prijatniji. Naravno, prvenstveno se misli na održavanje željene temperature unutar prostorije ali to nije jedini faktor koji treba regulisati. Kretanje vazduha kroz prostoriju kao i njegova higijenska ispravnost, bitni su kada je riječ o komforu, te shodno tome, različiti sistemi grijanja omogućavaju i različite komforne uslove.

Danas postoji niz različitih sistema, koji moraju biti podesivi, u zavisnosti od potrebne količine toplote i slično. Ono što karakteriše svaki od tih sistema, jeste izvor kojim se dobija ta toplota. Postoji više načina o kojima će nešto preciznije biti rečeno u samom radu. Jedan od najzastupljenijih, koji omogućava snadbjevanje jako velikog broja potrošača jeste daljinski sistem grijanja.

Karakteristika ovog sistema jeste da se proizvodnjom energije na jednom mjestu (toplane, solarne kolektorske elektrane, termoelektrane i sl.), putem sistema cijevi, kroz koje struji topao fluid potrošači snadbjevaju. Iz ovog kratkog opisa, jasno se vidi da sistem daljinskog grijanja ima niz prednosti i nedostataka koje će biti predstavljene u daljem radu.

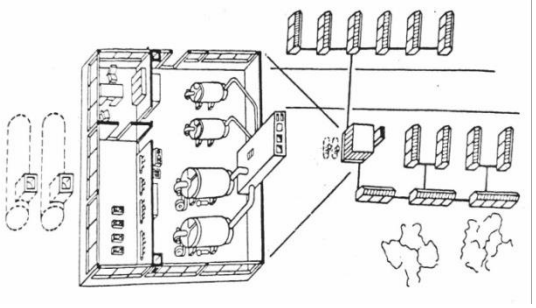
NAPOMENA:
Ovaj rad je proistekao iz master rada čiji je mentor bio doc. dr Aleksandar Anđelković.

2. SISTEMI GRIJANJA

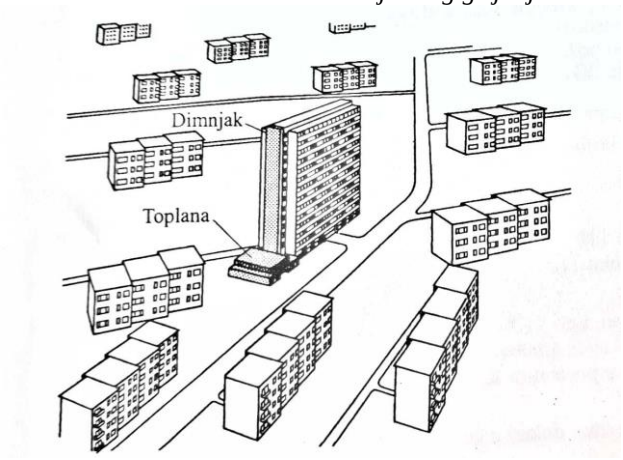
Sisteme grijanja delimo na dva osnovna tipa, individualni (grejno tijelo predstavlja zid kotla) i centralni (fluid se zagrijava te se sistemom cijevi šalje do ostalih grejnih tijela, na više lokacija).
U induvidualne sisteme ubrajamo kamine, kaljeve peći, gvozdene peći, gasne grijače itd.
Daljinskih tipova ima više, i o njima će biti više riječi u daljem tekstu.

3. CENTRALNI SISTEMI GRIJANJA

Kako bi se povećao komfor, smanjili troškovi grejanja i ostvario što manji loš uticaj na okolinu, došlo je do razvoja centralnog sistema grijanja koji proizvodnju energije, odnosno zagrijavanje fluida vrši na jednom mjestu, a zatim ga šalje na više različitih lokacija. Ovi sistemi su prošli niz modifikacija od nastanka do danas. Najzastupljeniji vid ovog sistema jeste daljinsko grijanje, koje se javlja u svim gradskim naseljima i karakteriše ga priprema prenosnog medijuma u kotlarnici, koji se putem sistema cijevi, podzemnim putem prenosi do potrošača.



Slika 1. *Blokovski sistem daljinskog grijanja*



Sl 2. *Daljinsko grijanje sa kotlanicom u sklopu najviše zgrade*

Dva su osnovna tipa daljinskog sistema:

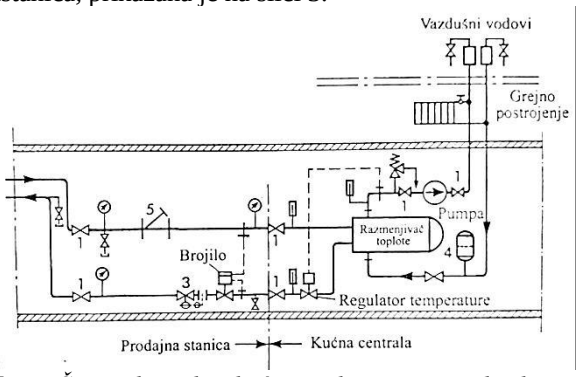
a) Blokovsko daljinsko grijanje (slika 1.)

b) Daljinsko grijanje sa kotlanicom u sklopu najviše zgrade (slika 2.)

Prema temperaturi fluida daljinski sistem grijanja se deli na:

c) vrelovodno(toplovodno) grijanje sa fluidom temperature do 110°C,
d) vrelovodni sistem daljinskog grijanja sa nosiocem toplote preko 110°C,
e) parno grijanje

Ono što je karakteristika prvog sistema sa temperaturom preko 110°C jeste da zahtjeva niz dodatnih troškova, koji se tiču kvaliteta materijala opreme koja se koristi, kao što su pumpe, zaptivači, itd. Najveća razlika se uočava u kućnim podstanicama. Zbog povišene temperature, a i pritiska koji se ostvaruje u ovom sistemu, čest slučaj je da podstanice imaju u sebi razmjenjivač toplote, čime je kućna instalacija odvojena od vrelovodne trase. Razmjenjivač prouzrokuje manju količinu predate energije kućnoj instalaciji, često servisiranje, čišćenje... Jedna takva podstanica, prikazana je na slici 3.



Slika 3. *Šematski prikaz kućne podstanice sa indirektnim priključkom na vrelovod(1-kugla ventil, 3-regulator protoka, 4-ekspanziona posuda, 5-hvatač nečistoća)*

4. MOGUĆNOSTI POSTAVLJANJA VRELOVODNE MEŽE

Kofiguracija terena, zahtjevi potrošača, narušavanje infrastrukture i mnogi drugi faktori mogu da utiču na to, koji od načina upotrijebiti kako bi se vod postavio kroz željenu relaciju. Osnovna podjela predstavlja podzemno i nadzemno vođenje koje se koristi u situacijama samo kada je neophodno ili kada se traži namjenski.

4.1. Postupak podzemnog vođenja

U zavisnosti od potreba, zahtjeva ili mogućnosti terena, podzemno vođenje trase se može ostvariti na sledeće načine:

f) postupak sa izolovanim cijevima kroz zemljani rov,
g) postavljanje trase kroz betnoske kanale, različitih oblika i namjene,
h) polaganje cijevi sa zalivanjem bitumenom,
i) postupci posebne izvedbe koji se ne sreću tako često.

4.2. Postupak nadzemnog vođenja

Ukoliko ne postoji ni jedan drugi način za vođenje, osim nadzemnim putem, pristupa se toj metodi. U njih ubrajamo:

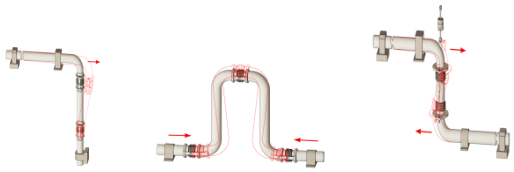
j) spoljni vodovi na betonskom podnožju
k) spoljni vodovi na nosačima
l) spoljni vodovi na mostovima ili nosećim nosačima
m) vodovi u podrumima
n) vodovi u garažama

5. OSNOVNI ELEMENTI CIJEVOVODA

Kako bi se fluid trasnportovao, od jedne do druge lokacije, neophodno je obezbijediti niz elemenata koji čine:

1. Cjevi-osnovni element, zadužen za trasnport. Izrađuju se u komadima dužine 3m sa izolacijom od poliuretanske tvrde pjene i sa zaštitnom cijevi,
2. Elementi zaduženi za promjenu pravca trase, lukovi, paralelni ogranci i slično (slika 4.)
3. Elementi zaduženi za kompenzaciju termičkih dilatacija („L“ „Z“ „U“ kompezatori) (slika 5.)

Slika 4. *Elementi zaduženi za promjenu pravca*



Slika 5. *Elementi zaduženi za kompenzaciju termičkih dilatacija*

Energetski pasoši za stambene i nestambene zgrade se sastoje od pet strana, dok energetski pasoš za ostale zgrade ima 3 strane.

6. PROJEKTOVANJE VRELOVODNE TRASE

Projektovanje vrelovodne trase predstavlja niz postupaka i aktivnosti koje je potrebno izvesti po adekvatnim standardizovanim normama. Prvenstveno treba prikupiti dovoljno podataka koji će omogućiti projektovanje. To podrazumijeva katastarske podloge kao i izlazak na teren i snimanje situacije.

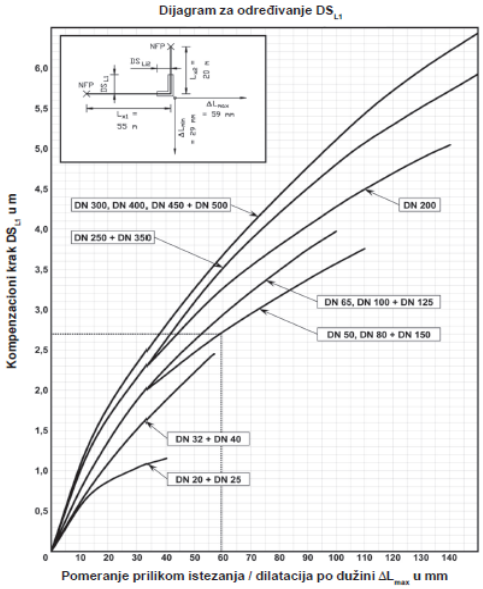
Na osonovu dobijenih energetskih saglasnosti i uspostavljanjem novoprojektovane trase, vrši se određivanje prečnika cijevovoda, njegovo ucrtavanje u katastarske podloge. Sve to je potrebno poslati javnim komunalnim preduzećima kako bi se ustanovilo da se trasa može izvesti po željenoj putanji. Na osnovu dobijenih saglasnosti, moguće je pristupiti projektovanju.

6.1. Statički proračun

Zbog visoke temperature fluida unutar cijevovoda, dolazi do širenja materijala. To širenje je potrebno izračunati i kompezovati ga elastičnim elementima. Ovaj proces se može odrediti na različite načine. Upotrebom obrasca (1) dobija se dužina dilatacije.

Na osnovu nje i prečnika cijevi dolazi se do dužine kompenzacionog kraka prema slici 6. ukoliko se radi o „L“ kompenzatoru. Ista procedura se vrši i za druge kompenzatore, s tim da je kriva dijaagrama drugačija.

$$\varepsilon_{\mathrm{L}} = \alpha \cdot L_x \cdot \Delta T \qquad (1)$$



Slika 6. *Određivanje dužine kompenzacijskog kraka „L" kompenzatora*

U slučaju se radi o velikoj trasi, koja ima veliki broj dionica, koje je potrebno kompenzovati, ovaj proračun je mnogo lakše odraditi u nekom od sofvera koje imaju te mogućnosti, kao što su ISO Plus, LOGSTOR, SIS KMR i drugi. Ovim softverima, moguće je dobiti sve podatke koji se tiču statike, kao što su aksijalni i tanagencionalni naponi materijala cijevi, starenje, uticaj debljine rova, itd.

6.2. Hidraulički proračun

Pri strujanju fluida unutar cijevi vrelovoda, dolazi do pada pritiska. Taj pad pritiska je prouzrokovan trenjem kroz cijevovod kao i lokalnim otporima nastalih zbog raznih elemenata poput lukova, redukcija, odvajanja tokova i slično. Hidraulički proračun se može prikazati kroz niz obrazaca dazih ispod:

$$Q = V \cdot \rho \cdot c_p \cdot \Delta t \quad (2)$$

$$V = \frac{Q}{\Delta t \rho \cdot cp \cdot} \quad (3)$$

$$w = \frac{V}{A} \quad (4)$$

$$\Delta p = \Delta p_{lok} + \Delta p_{lin} \quad (5)$$

$$\Delta p_{lin} = \frac{W^2 \cdot \lambda \cdot L \cdot \rho}{2 \cdot du} \quad (6)$$

$$\Delta p_{lok} = \frac{W^2 \cdot \rho \cdot \Sigma \xi}{2 \cdot du} \quad (7)$$

$$\lambda = 0.11 \cdot \sqrt[4]{u : du} \quad (8)$$

Q - količina energije koju dionica treba da prenese [kW]
V - zapreminski protok [m^3/s]
ρ - gustina tečnosti na temperaturi potisa [kg/m^3]
c_p - specifična toplotna kapacitivnost vode [kJ/kg]
Δt - temperaturska razlika potisa i povrata [°C]
w - brzina strujanja fluida unutar cijevi [m/s]
A - površina unutrašnjeg poprečnog presijeka [m^2]
Δp - ukupan pad pritiska [Pa]
Δp_{lok} – pad pritiska usled lokalnih otpora [Pa]
Δp_{lin} - pad pritiska usled linijskih otpora [Pa]
λ - koeficijent trenja [/]
u - apsolutna hrapavost [mm]
du - unutrašnji prečnik [m]
$\Sigma \xi$ - zbir lokalnih otpora na posmatranoj dionici [/]

7. PRIMJER PROJEKTA VRELOVODNE TRASE

U narednom primjeru koji predstavlja rekonstrukicju postojeće vrelovodne trase u Dunavskoj ulici, i ulici Ive Lole Ribara, biće dat idejni projekat sa priloženim statičkim proračunom i grafičkom dokumentacijom neophodnom za izradu projekta.

7.1. Projektni zadatak

Potrebno je dati projektno-tehničku i investicionu dokumentaciju za rekonstrukciju vrelovoda od ulice Ignjata Pavlasa, kroz Dunavsku a zatim kroz ulicu Ive Lole Ribara. Takođe, ovim projektom je predviđeno rekonstruisanje vrelovoda u bulevaru Mihajla Pupina broj 28. Projekat je potrebno izraditi u skladu sa Zakonom o planiranju i izgradnji, odnosno pravilnikom o sadržini i načinu izrade tehničke dokumentacije.
Projektni uslovi koje je neophodno ostvariti:

- Postojeći magistralni vrelovod za priključenje novoprojektovane trase je dvocijevni i izveden je od predizolovanih cijevi, pa je će se i nova trasa izvoditi od istih cijevi,
- Potrebne količine toplote, koje je potrebno prenijeti u novoprojektovanoj trasi, i na osnovu koje će se izvršiti dimenzionisanje cijevi uzeti iz izdatih energetskih saglasnosti za pojedine objekte,
- pritisak na mjestu početka vrelovodne trase usvojiti da je 7,2 bara, dok u povratnom vodu pritisak ima vrijednost od 2 bara.
- Potrebno je usaglasiti ukrštanje vrelovoda sa ostalim instalacijama

Pri izradi projekta, neophodno je pridržavati se normi propisanih standardnom EN13941.

7.2. Tehnički opis

Ovim projektom je obuhvaćena Rekonstrukcija vrelovoda od Ignjata Pavlasa u bloku ulica Dunavska i Ive Lole Ribara u Novom Sadu, k.p. 9398, 9399 K.O. Novi Sad I i k.p. 7730, 75, 76, 77, 78/2, 100, 79, 102, 96, 97/1, 83, 80, 98, 111, 92, 101, 107/2, 108/1, 107/1, 74, 7732/1, 140 K.O. Novi Sad II. Projektovano je u skladu sa Zakonom o planiranju i izgradnji, odnosno Pravilnikom o sadržini i načinu izrade tehničke dokumentacije, Pravila o radu distributivnog sistema JKP „Novosadska Toplana".

Idejni projekat je urađen na osnovu Prethodne saglasnosti i kopije plana vodova koju je izradio Republički geodetski zavod iz Novog Sada.

Novoprojektovana trasa se nadovezuje na postojeći vrelovod, na uglu Dunavske i ulice Ignjata Pavlasa, odakle se ispod pješačke staze vodi do prve podstanice u Dunavskoj 31. Novoprojektovana trasa se pruža duž Dunavske ulice ka ulici Ive Lole Ribara gdje skreće normalnim lukom L2. Dalje, novoprojektovana trasa se vodi čitavom dužinom u ulicu Ive Lole Ribara, do podstanice u objektu br. 26-28 na Bulevaru Mihajla Pupina. Ono što je važno naglasiti, jeste da se trasa koja prolazi ispod bulevara Mihajla Pupina ne rekonstruiše, jer rekonstrukcija prethodno odrađena zbog sanacije kvara.

U ulici Ive Lole Ribara, paralelnim ogrankom Po5 DN250/100 novoprojektovana trasa skreće u ulicu Vojvode Putnika, napajajući 7 podstanica.

7.3. Statički proračun

Sataički proračun je moguće odraditi ručno, primjenjujući sve formule date u poglavlju 6.1. S obzirom da je riječ o velikom projektu, što se može viditi i u grafičkoj dokumentaciji, za proračun statike se upotrebljavaju dva softvera:

a) ISOPLUS, kojim se proračunava statičko opterećenje „L“, „Z“, i „U“ kompenzatora,
b) LOGSTOR, kojim se proračunava statičko opterećenje paralelnih i „T“ ogranaka.

7.4. Hidraulički proračun

O detaljima hidrauličkog proračuna, obrascima koji se koriste pri njegovom izvršenju te vrijednostima koeficijenata i ostalo, razmatrano je u poglavlju 6.2. Ako se uzme u obzir da sve potrebne veličine koje se koriste pri hidrauličkom proračunu imaju funkcionalnu zavisnost, jednostavnim povezivanjem ćelija u Microsoft Excelu, veoma lako se može doći do pada pritiska unutar mreže.

Pad pritiska za najudaljeniju tačku novoprojektovane trase dat je u tabeli 1.

Tabela 1. *Hidraulički proračun pada pritiska*

				$t_{u=}$	140 °C											
	Δt=	70 °C		$t_{iz=}$	70 °C											
	c_p=	4,187	KJ/kg°K													
Br.	V	t	Qu	ρ(t)	DN	d_s	s	d_u	*w*	R	L	R x L	ξ	Z	RL+Z	Δp
od	l/s	°C	W	kg/m³		mm	mm	m	m/s	Pa/m	m	Pa		Pa	Pa	Pa
1	62,1114	105	17.384.120	954,95	250	273	5	0,263	1,144	30,9	13,0	402	1	625	1027	1027
2	62,0207	105	17.358.740	954,95	250	273	5	0,263	1,142	30,8	18,0	555	0,5	311	867	1894
3	59,4948	105	16.651.780	954,95	250	273	5	0,263	1,096	28,5	194,0	5525	6,5	3726	9251	11144
4	55,9219	105	15.651.780	954,95	250	273	5	0,263	1,030	25,3	28,0	708	1	506	1215	12359
5	54,7164	105	15.314.380	954,95	250	273	5	0,263	1,008	24,3	26,0	631	1	485	1116	13475
6	49,3037	105	13.799.420	954,95	250	273	5	0,263	0,908	19,9	61,0	1213	2,5	984	2197	15672
7	47,8238	105	13.385.240	954,95	250	273	5	0,263	0,881	18,8	26,0	488	2,5	926	1414	17086
8	42,6238	105	11.929.810	954,95	200	219,1	4,5	0,2101	1,230	47,1	19,0	895	1	722	1617	18703
9	41,4454	105	11.600.000	954,95	200	219,1	4,5	0,2101	1,196	44,6	82,0	3658	2	1366	5025	23727
10	1,2128	105	339.440	954,95	32	42,4	2,6	0,0372	1,116	340,3	67,0	22800	2,5	1488	24288	48016

8. ZAKLJUČAK

Sistem daljinskog grijanja je jedno od najčešćih rješenja snadbjevanja objekata toplotom, naručito kada je riječ o većima naseljima, odnosno gradovima. Sama karakteristika ovog sistema, da se dobijanje energije ostvaruje na jednom mjestu, čime se smanjuju troškovi namjenjeni za kotlarnice, zatim smanjenje zagađenja i slično, govore o pogodnostima koje ovaj sistem sa sobom nosi.

Prvobitne vrelovodne mreže koristile su cijevi sa lošom izolacijom, a nerijetko se sreće slučaj da se ispod površine zemlje nađu cijevi vrelovoda koje nisu obložene nikakvom toplotnom izolacijom. Upravo takva situacija predstavlja ogroman prostor za uštedu, jer se transport fluida kroz neizolovanu cijev ispostavio kao totalno neefikasan. Ovim se daje na značaju da rekonstrukcija vrelovodne mreže ne predstavlja ništa drugo nego finansijska ulaganja koja će se brzo isplatiti uz povećanje kvaliteta sistema grijanja i mogućnosti usavršavanja rada toplane.

9. LITERATURA

[1] Recknagel H, Sprenger E, Schramek E-R, Čeperković S, Grejanje i Klimatizacija, Interklima, Vrnjačka Banja, 2012.
[2] Anđelković A, Grejanje, ventilacija i klimatizacija, Faklutet tehničkih nauka, Novi Sad, 2017.
[3] Kozić Đ, Vasiljević B, Bekavac V, Priručnik za termodinamiku, IRO „Građevinska Knjiga“ Beograd, 1983.
[4] Hansjoachim M, Dietrich W, Selecting Centrifugal Pumps, KSB Aktiengesellschaft, Germany
[5] ISOPLUS, dostupno na: https://isoplus.rs/, [datum pristupa: 10.09.2019.]
[6] Logstor, dostupno na: https://www.logstor.com/, [datum pristupa: 15.09.2019.]

Kratka biografija:

Nikola Bartula, rođen u Sokocu, BiH RS 1995. godine.
Fakultet tehničkih nauka, Novi Sad
Mašinstvo-Energetika i procesna tehnika
Osnovne studije završio 2018. godine

Email: galanikola95@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 628.8
DOI: https://doi.org/10.24867/07AM08Berec

EKSPERIMENTALNA I NUMERIČKA ANALIZA TERMIČKOG KOMFORA U STAMBENOM OBJEKTU

EXPERIMENTAL AND NUMERICAL ANALYSIS OF THERMAL COMFORT IN A RESIDENTAL BUILDING

Gabor Berec, Aleksandar Anđelković, *Fakultet tehničkih nauka, Novi Sad*

Oblast- MAŠINSTVO

Kratak sadržaj –. *U radu je urađeno merenje i analiza pokazatelja termičkog komfora. Merenje uključuje i eksperimentalnu i numeričku analizu. Parametri koji su bili predmet analize su temperatura po suvom termometru, brzina strujanja vazduha i relativna vlažnost u prostoriji. Pored merenja navedeni su osnovni principi numeričke analize i teorijske osnove računarske dinamike fluida. Urađen je i opis mernih uređaja, koji su korišćeni za eksperimentalnu analizu sa detaljnim objašnjenjem pravila merenja prema standardima. Dobijeni rezultati su detaljno protumačeni i uređena je uporedna analiza.*

Ključne reči: *Eksperimentalna analiza, numerička analiza, klimatizacija, termički komfor*

Abstract – *This paper involves the analysis and measurements of thermal comfort indicators. The paper includes both experimental measurements and numerical analysis in the form of a computational fluid dynamics. The measured parameters were the dry bulb temperature, the velocity of the airflow, mean radiant temperature and the relative humidity in the room. In addition to measurements, the basic principles of numerical analysis and the theoretical basis of computational fluid dynamics had been interpreted. A description of the measuring devices, which were used for the experimental part, was also made with a detailed explanation of the measurement method according to the standards. The measurement results were interpreted in detail and a comparative analysis was made.*

Keywords: *Experimental analysis, numerical analysis, climatisation, thermal comfort*

1. UVOD

Kao nastavak istraživanja u mom bachelor radu, sada će se uvesti i praktično merenje uslova termičkog komfora u prostoriji paralelno sa računarskim modelovanjem i CFD analizom. Navešće se važniji segmenti metodologije merenja u klimatizaciji i praktične, korisne načine na koje merenja mogu da se izvedu.

Biće reči i o samim uređajima, pomoću kojih se meri, o njihovim karakteristikama i opsezima merenja. U parametre, koje će se meriti spada brzina strujanja vazduha, temperatura, i relativna vlažnost.

Merenje navedenih veličina će se izvesti u više tačaka raspoređenih u prostoru, sa posebnom pažnjom na poziciju gde osobe borave.

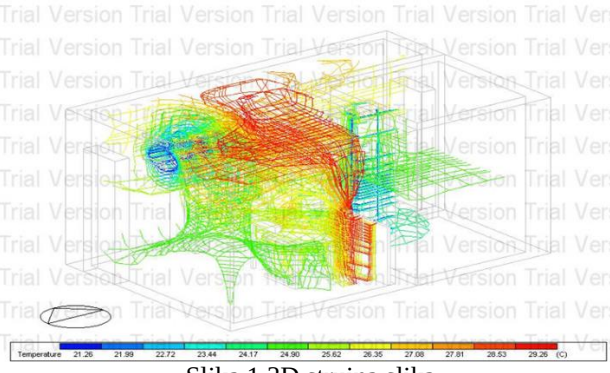
Pored praktičnih merenja u prostoriji podrazumeva se i izrada 3D modela prostorije u softverskom paketu i analiza spomenutih komfornih uslova u virtualnom prostoru. Za računarsku simulaciju, koristiće se podešavanja, koji su najbliži uslovima dana, kad su se vršila praktična merenja (temperatura, osunčanost, vlažnost vazduha, brzina vetra).

Za prezentaciju rezultata, poslužiće slike dobijene iz CFD analize, kao i rezultati dobijenih brzina i temperatura ubačenog vazduha, koje zadovoljavaju korisnike. Takođe će se priložiti rezultati stvarnih merenja sa mernim instrumentima. Iz ovih baza podataka se lako može dokazati da u pojedinim tačkama prostora (prvenstveno se misli na mesta gde ljudi borave) koliko se razlikuju izmereni podaci i izračunati podaci iz softvera.

2. NUMERIČKA ANALIZA

Računarska dinamika fluida je osnovni alat za numeričku analizu u klimatizaciji, a u našem slučaju se koristi softverski paket Design Builder. Ovaj program poseduje mogućnost pravljenja 3D modela ili importovanje istog iz nekog naprednijeg softvera, kao što je Revit.

U CFD analizi radi uštede na računskoj snazi i vremenu potrebnog za proračune, koriste se pojednostavljeni geometrijski oblici distributivnih rešetki za vazduh i unutrašnji objekata u prostoriji, kao nameštaj i telo čoveka. Na sledećoj slici se može videti trodimenzionalna slika raspored toplijeg i hladnijeg vazduha u prostoriji. Naglasak sa konturama dobija vazduh, koji se kreće određenom brzinom a ne miruje. Jasno se može videti na slici položaj toplotnog opterećenja i klima uređaja (crvena i plava boja respektivno).

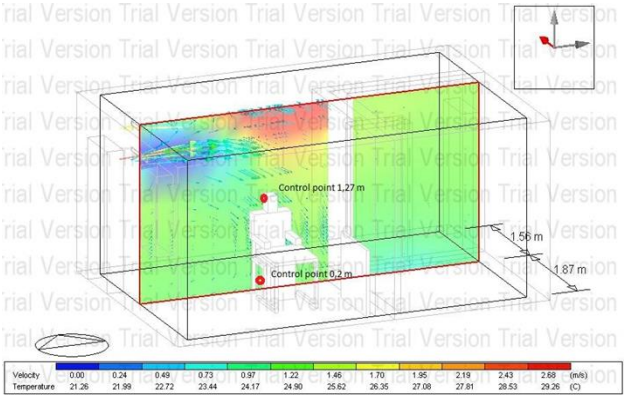


Slika 1 3D strujna slika

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

Na sledećoj slici se može videti položaj čoveka u sedećem položaju i sa položajem kontrolne tačke pri analizi. Prva tačka je na 1,27m a druga tačka na 0,2m od poda.

Na razmaku između ove dve tačke može da se ustanovi da li postoji vertikalna razlika temperature, koja utiče negativno na osećaj termičkog komfora u prostoriji.



Slika 2 Položaj kontrolnih tački u modelu

3. OSNOVNI PRINCIPI EKSPERIMENTALNOG MERENJA

- Da bi se u jednoj prostoriji, izmerili termički pokazatelje unutrašnje klime merenja moramo izvoditi na mestima, gde ljudi provode značajan deo svog vremena, sedeći ili stojeći. Za definisanje mernih mesta i načina, na koji se izvode merenja poslužiće se prevod delova ASHRAE Thermal Comfort Standard 55 iz 2017.:

Za merenje se uzimaju mesta u prostoriji na kojima je najveća verovatnoća da će se ljudi boraviti. U slučaju da se ne mogu odrediti takva mesta, uzima se geometrijska sredina te prostorije ili

1 metar od centra svakog zida prostorije. U slučaju da je reč o spoljnom zidu sa prozorom/prozorima merenje se pozicionira 1 metar od centra najvećeg prozora.

Merenja se moraju izvesti dovoljno udaljeno od granice zona boravka.

Apsolutnu vlažnost je dovoljno izmeriti u jednoj tački u prostoriji, dok relativna vlažnost mora da se meri u svim tačkama.

Visina mernih mesta za temperaturu i brzinu strujanja, mereno od poda:

- 0,2 m
- 0,6 m
- 1,27 m

Za dobijanje prosečne brzine strujanja vazduha, potrebno je merenja izvršiti na svakih 3 minuta u mernom periodu.

Za dobijanje veličinu fluktuacije unutrašnje temperature ili dobijanje prosečne temperatura u mernom periodu, potrebno je merenja izvršiti na svakih 5 minuta u mernom periodu, i pomoću jednačine za intenzitet promene (stepeni po satu). Intenzitet promene

$$Intenzitet\ prom.\left(\frac{^{O}C}{h}\right) = 60 \cdot \frac{t_{0,max} - t_{0,min}}{vreme\ (min)}$$

U podacima merenja treba navesti i sledeće podatke:

Količina vazduha, koja se dovodi mehaničkim putem u prostoriju – u slučaju da postoji mehanički sistem za ventilaciju, razlika unutrašnje i temperature rashladnog vazduha, brzina ulazne vazdušne struje, mesto i veličina izlazne rešetke, vrsta ventilacionog sistema, temperatura graničnih površina prostorije, temperature polazne i povratne vode u rashladnom sistemu.

4. EKSPERIMENTALNO MERENJE

Za izvođenje merenja termičkih uslova komfora u datoj prostoriji, bilo je potrebno formirati jednu mernu stanicu, koja se postavlja na poziciju, gde će se prema pretpostavkama boraviti osoba. Vrednosti koji će se meriti su temperatura, brzina strujanja vazduha, relativna vlažnost vazduha, prosečna temperatura ozračenja

U ove svrhe izabrani su sledeći uređaji:

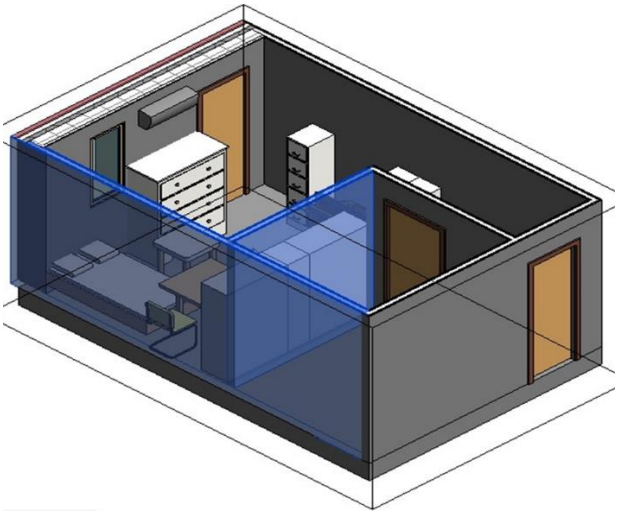
•Testo 435 – uređaj za merenje temperature i brzine strujanja vazduha (na principu užarene niti)

•Testo 177 H1 – uređaj za merenje vlažnosti vazduha i temperature.

•Trotec BP 15 – infracrveni termometar

•Arduino UNO CH430 R3 sa termistorom tip VU NTC 100k 3950 koja je smeštena u crnoj lopti za stoni tenis i simulira apsolutno crno telo, koja upija svo zračenje okolnih površina.

Prostorija, koja će biti predmet ove analize je korisne površine 23,8 m2, i sa 3 strane je zaokružena grejanim unutrašnjim zidovima, a sa jedne strane se nalazi otvorena terasa i izolovani spoljni zid. Prostorija je u dovoljno velikoj meri zatvorena i otvori prema spoljnoj sredini se nalaze samo sa jedne strane.

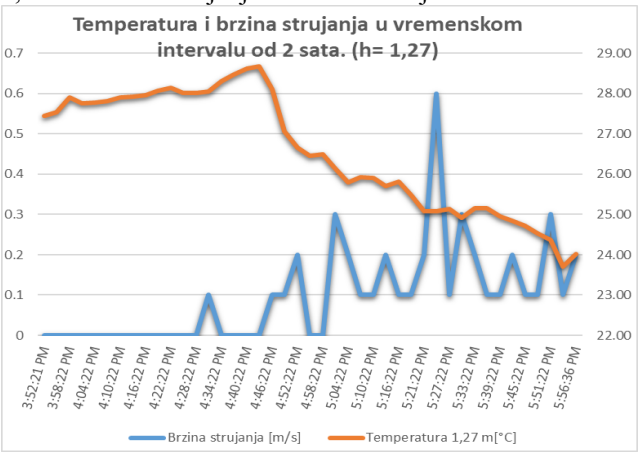

Slika 3 Prostor, koji se analizira (Revit model)

Spoljni zidovi prostorije su građeni od termo bloka sa izolacijom od stiropora debljine 12 cm. Prozori i vrata su PVC sa jednokomornim staklima. Podaci ovih konstrukcija su unešeni i u softverska podešavanja, da bi se dobili odgovarajući polazni podaci koeficijenta prolaza toplote (tzv. U-value). Kao spoljašnji vremenski uslovi zadati su 'Weather File' za Novi Sad sa merne stanice Rimski Šančevi.

Merenja su se izvodila na posebno postavljenom mernom štandu i fiksirani su na potrebne visine.

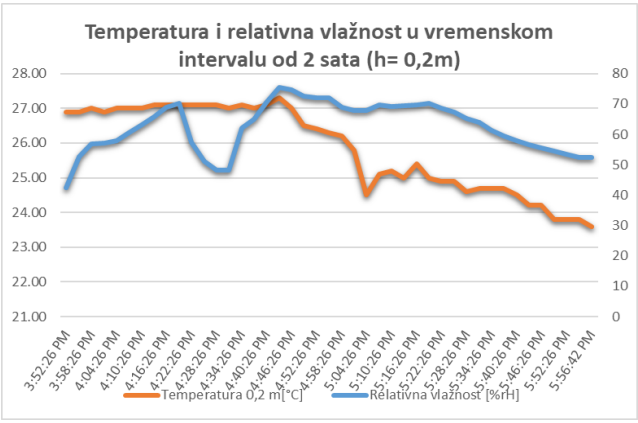
Slika 4 Raspored mernog štanda

Rezultati merenja pomoću uređaja Testo 435. To su temperatura po suvom termometru, izmerena na visini od 1,27 m i brzina strujanja vazduha u istoj tačci.



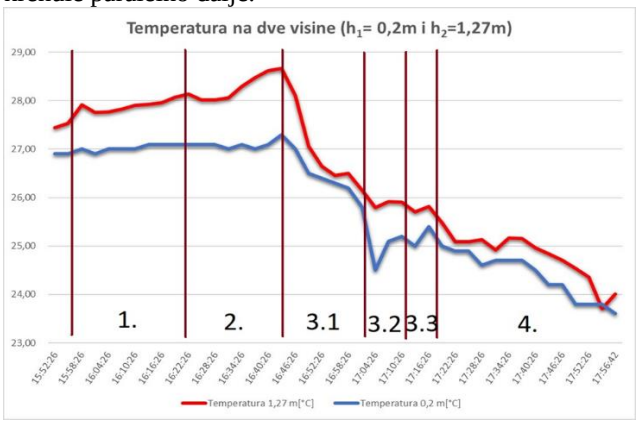
Dijagram 1 Rezultati dobijeni na Testo 435

Rezultati na sledećem dijagramu su dobijeni pomoću uređaja Testo 177. To su temperatura po suvom termometru izmerena na visini od 0,2 m i relativna vlažnost vazduha.



Dijagram 2 Rezultati dobijeni na Testo 177-H1

Posmatrajmo dijagram promene temperature u toku vremena, na kojem dosta tačno mogu se odrediti tačke kada su se dešavale promene. Na grafikonu su dve linije, koje označavaju dve različite visine senzora. Ako povučemo vertikalne linije, možemo prepoznati kad su otvorena spoljna vrata i kad je uključen klima uređaj. Dok je temperatura rasla i nije bio uključen klima uređaj, temperaturno polje je delimično bio stratifikovan. To znači da na osnovu razlike u gustinama toplijeg i hladnijeg vazduha definišu se slojevi. Tako je moguće videti da između linije postoji veći razmak do te tačke. Tokom prirodne ventilacije u početku je temperatura krenula naviše, nažalost se ovaj proces prirodne ventilacije je završen posle 10 minuta i uključilo se klima uređaj. Ova akcija je takođe jasno vidljiva i na dijagramu takođe, pošto je temperatura posle uključenja klima uređaja, jasno krenula da pada,i šta je još važnija, slojevi vazduha su krenuli da se pomešaju. Ovo je uticalo na razliku temperature koja je sada smanjena i linije su krenule paralelno dalje.



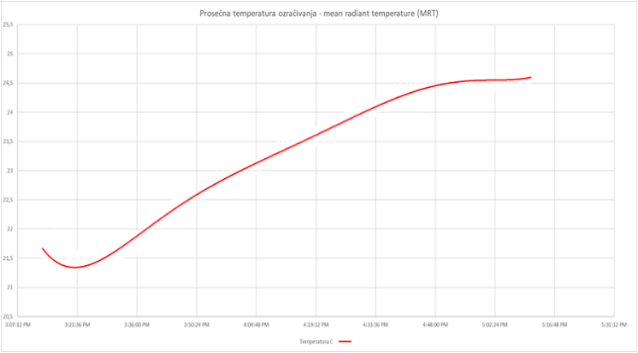
Dijagram 3 Veza temperatura na dve visine

Prosečna temperature ozračenja je veoma značajan faktor u određivanju termičkog komfora jedne prostorije. Na primer u letnjem periodu, kada se prostorije hlade, potrebno je da prođe određeni period dok se pored unutrašnjeg vazduha, ohladi se i termalna masa u prostoriji. Pod termalnom masom se misli na zidove, podove, plafone, okolne površine, nameštaja i ostalih čvrstih tela u prostoriji.

Za merenje prosečne temperature ozračenja, služio je Arduino UNO sa odgovarajućim termistorom I stalkom za merenje. Ofarbana loptica simulira apsolutno crno telo, koja upija svo zračenje u prostoriji i preuzima odgovarajuću temperaturu okolnih površina. Termistor se nalazi u loptici za stoni tenis koja je fiksirana na stalak i podignuta je na visinu čovekove glave pri sedenju. Postavljani merni štand je prikazan na slici 5.

Slika 5 Merni štand za određivanje MRT

Da bi se dobila stvarna vrednost prosečne temperature ozračenja potrebno je pratiti očitane vrednosti i crtati dijagram od ovih vrednosti. Kad kriva prelazi u horizontalan, ili bar približno horizontalan položaj, merenje s može zaustaviti. To znači da se crna kugla u potpunosti prilagodila unutrašnjim uslovima. To je važno i u slučaju da se pre eksperimenta kugla nalazila na ekstremno toplom ili ekstremno hladnom mestu.
Dalje oscilacije temperature se mogu tumačiti kao promene spoljašnjih uslova u toku vremena. Brzo zaoblačenje napolju ili promena unutrašnjih uslova može da izazove dalje promene u dijagramu prosečne temperature ozračenja u vremenu. Na kraju merenje dobila se sledeća kriva:



Dijagram 4 Prosečna temperatura ozračenja

Rezultat merenja je da prosečna temperatura ozračenja iznosi: 24,53 ^{o}C. Pri tom unutrašnja temperatura po suvom termometru iznosila: 25,3 ^{o}C. Spoljna temperature je bila: 25,6 ^{o}C (sunčano). Relativna vlažnost iznosila: 65%
Napomena:
Korišćeni termistor NTC 100k 3950 za određivanje prosečne temperature ozračenja nije baždaren niti je optimalno kalibrisan, moramo uzeti u obzir grešku pri merenju koja može da iznosi i do +/-1 oC prema fabričkim podacima ovih termistora.
Pri merenju vrednosti se variraju na svakih 5-10 očitavanja za maksimalnih 0,4-0,5 ^{o}C. Iz ovih razloga prikazana kriva je aproksimacija dobijenih rezultata pomoću polinoma 6. stepena.
Dobijeni podaci se unesu u kalkulator termičkog komfora, univerziteta California Berkley, koja računa pomoću standarda ASHRAE 55 iz 2017. i EN15251 iz 2006.
Dodatno se uvrsti nivo oblačenosti “clo” i nivo metaboličke aktivnosti “met”, koje iznose:
Metabolička aktivnost= 1 met (sedeći položaj, mirno)
Oblačenost=0,61 (bluza dugih rukava, duge pantalone), dobijaju se sledeći rezultati ocene termičkog komfora:
Prema ASHRAE 55 PPD=5% , PMV= -0,05 koja odgovara kriterijumima ovog standarda. Prema EN15251 PPD=5%, PMV=-0.07.i tako spada u prvu kategoriju termičkog komfora.

5. ZAKLJUČCI

Pri upoređivanju rezultata dobijenih iz eksperimenta i pomoću računarske analiza, najznačajniji je zaključak da trajanje eksperimenta treba da bude duži. Način izvođenja ovog eksperimenta iziskuje da se instrumentalno merenje obavlja prvo iz razloga što spoljne uticaje (kao što su osunčanost, temperatura, vlažnost vazduha, brzina vetra) je nemoguće pogoditi unapred.
Instrumentalno merenje je bilo vremenski ograničeno, međutim pokazatelji su se vidljivo i jasno menjali usled različitih promena u kontrolnom prostoru. Posle toga su se radile računarske simulacije koje za svaki period merenja su pokazali veću vrednost od instrumentalnog merenja u slučaju da se temperatura povećava ili manju vrednost u slučaju da se temperatura smanjila. Glavni razlog ovog odstupanja je u vremenskom periodu instrumentalnog merenja. Za dobijanje rezultata koja su bliže rezultatima dobijenih pomoću simulacija potrebno je da instrumentalno merenja traje dosta duže da bi se akumulirana toplota u unutrašnjim objektima takođe rasipala. Akumulaciona masa koja se nalazi u prostoru, produžava odziv sistema i čini ga inertnim.
Ova činjenica stvara još jednu otežavajuću okolnost pri izvođenju instrumentalnog dela merenja. Ako se merenje radi dovoljno dugo, da bi se dobili ustaljeni vrednosti za temperaturu i vlažnost vazduha, spoljni uticaj će se promeniti što iziskuje ponovno menjanje unutrašnjih uslova. Na primer naoblačenje, neočekivano pojačanje vetra, promena temperature, kiša. Za izvođenje eksperimenta treba odabrati vremenski period u toku dana kada se očekuje najmanja promena spoljnih uticaja. Prilikom odabira ovog perioda može se oslanjati na kratkoročnu vremensku prognozu za taj dan.

6. LITERATURA

[1] http://comfort.cbe.berkeley.edu/, 22.10.2019.
[2] Standard Well V2 for buildings 2018
[3] Standard EN 15251:2006 Indoor environmental input parameters for design and assessment of energy performance of buildings addressing indoor air quality, thermal environment, lighting and acoustics.
[4] Standard BS EN 16798-1:2019 Energy performance of buildings – Ventilation for buildings
[5] ASHRAE Standard 55-Thermal Environmental Conditions for Human Occupancy. 2017.
[6] Mohammad H. Hosni, Ph. D.,Byron W. J, Ph.D, Hanming Xu. 1999. Experimental Results for Heat Gain and Radiant/Convective Split from Equipment in Building. ASHRAE Journal

Kratka biografija:

Gabor Berec rođen je u Novom Sadu 1993. god. Završio Tehničku Školu u Adi 2012. godine, iste godine upisao osnovne akademske studije na Fakultetu tehničkih nauka iz oblasti Mašinstva – Energetika i procesna tehnika. Odbranio diplomski rad na temu Uporedna analiza metoda distribucije vazduha pri klimatizaciji 2016. god. Trenutno student master studija na smeru Energetika i procesna tehnika.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 620.9
DOI: https://doi.org/10.24867/07AM09Tojaga

ENERGETSKA SERTIFIKACIJA OBJEKATA U SRBIJI

ENERGY CERTIFICATION OF BUILDINGS IN SERBIA

Željko Tojaga, *Fakultet tehničkih nauka, Novi Sad*

Oblast- MAŠINSTVO

Kratak sadržaj – *U radu su prikazane teorijske osnove energetske sertifikacije objekata u Srbiji. Izvršena je energetska sertifikacija objekata kao i uporedna analiza energetskih potreba u zavisnosti od sistema grejanja.*

Ključne reči: *Energetska efikasnost, zakonodavni okviri, energetska sertifikacija objekata*

Abstract – *This paper presents theoretical bases for energy certification of the buildings in Serbia. The energy certification of the buildings was performed, as well as the analysis of energy needs, depending on the heating system.*

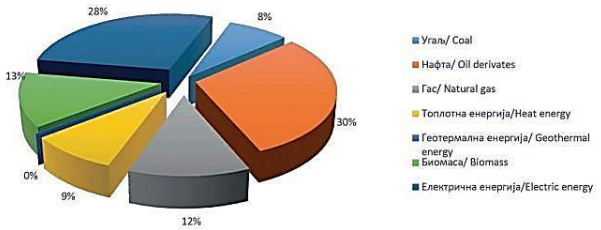
Keywords: *Energy efficiency, legislative framework, energy certification of buildings*

1. UVOD

Pojam energetske efikasnosti u praksi može se razmatrati na dva načina. Prvi način jeste da se ovaj pojam energetske efikasnosti odnosi na tehničke uređaje, dok drugi način jeste definisanje mera i ponašanja. U zgradarstvu energetska efikasnost predstavlja iskorišćenje manje količine energije za obavljanje nekog posla ili određene aktivnosti. Kod zgrada potrošnja manjih količina energije za zadovoljenje životnih potreba, a samim time se mora održati održavanje ugodne temperature, neophodnih uslova osvetljenja kao i drugih potreba za boravak ljudi u zatvorenom prostoru. Ukoliko se zgrada posmatra kao jedan sklop, dobro termički izolovani objekat manje troši energije za zagrevanje zimi i za hlađenje leti, a boravak ljudi u njoj je udobniji i kvalitetniji. Sa unapređenjem energetske efikasnosti u zgradarstvu doprinosimo globalnoj zaštiti životne sredine i smanjenju emisije štetnih gasova, koja nastaju sagorevanjem energenata koja koriste za zagrevanje prostora.

2. PREGLED STANJA U SRBIJI

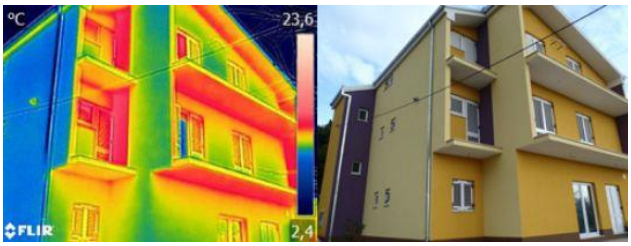
Potrošnja energije u zgradama koje se nalaze u Republici Srbiji u stalnom je porastu tokom poslednje decenije, tako da zauzima jako veliki udeo u ukupnoj bruto potrošnji energije. Ukupna potrošnja finalne energije dostigla je vrednost od 8,19 Mtoe u 2013. godini i to u sledećim sektorima potrošnje: domaćinstva, komercijalni sektor, industrija i transport, zgrade javne namene. Od energenata u potrošnji finalne energije dominantna je nafta sa 30 % i električna energija sa 28 %, nakon toga sledi ugalj sa 8 %, prirodni gas sa 12 %, toplotna energija sa 9 %, dok obnovljiva energija učestvuje sa 13%. Slika 1. predstavlja učešće energenata u ukupnoj finalnoj energiji u Republici Srbiji 2013.godine.



Slika 1. *Učešće energenata u ukupnoj finalnoj energiji u Republici Srbiji u 2013. godini*

3. POSTOJEĆA SITUACIJA U SRBIJI U ZGRADARSTVU

Veći broj zgrada u Republici Srbiji jeste neefikasan, naročito u zgradarstvu. Što se tiče zgrada, potrošnja energije predstavlja 36% od ukupne potrošnje u državi. Pored toplotne izolacije zgrade, najosetljiviji deo objekta jesu elementi stolarije-prozori i spoljašnja vrata. Što se tiče stolarije,koja je korišćena u objektima koji su bili izrađeni tokom prošloga veka jesu bili drveni ramovi sa jednostrukim staklom.
Na slici 2. prikazan je objekat koji je snimljen uz pomoć termovizijske kamere.



Slika 2. *Prikaz zgrade snimljenje uz pomoć termovizijske kamere*

Termovizija jeste beskontaktna metoda merenja temperature i njena raspodela je na površini posmatranog objekta.

4. PREGLED ZAKONODAVNIH OKVIRA U OBLASTI ENERGETSKE EFIKASNOSTI

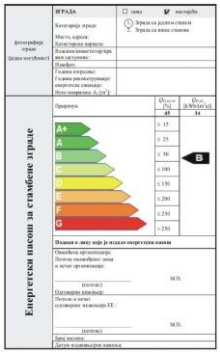
Proces energetske sertifikacije objekata u Srbiji je započet 30. septembra 2012. godine, kada je primena pravilnika iz oblasti energetske efikasnost zgrada postala obavezna. Pravilnik o energetskoj efikasnosti zgrada bliže propisuju zahtevana energetska svojstva, definisanje same metodologije proračuna termičkih svojstava zgrada, kao i određivanje zahteva za nove i postojeće objekte. Pravilnik o uslovima, sadržini i načinu izdavanja sertifikata o ener-

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

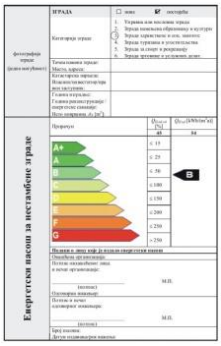
getskim svojstvima zgrada uređuje proces energetske sertifikacije zgrada, načinu izdavanja i sadržaj sertifikata, i definiše energetske razrede za stambene i nestambene zgrade, i to nove i postojeće.

4.1. Izgled i sadržaj obrazaca energetskih pasoša

Kao što je već rečeno postoje tri obrasca energetskog pasoša koji su prikazani u Pravilniku o uslovima, sadržaju i načinu izdavanja sertifikata o energetskim svojstvima zgrada.



Slika 3. Prva strana energetskog pasoša za stambene zgrade



Slika 4. Prva strana energetskog pasoša za nestambene zgrade

Slika 5. Prva strana energetskog pasoša za ostale zgrade

Energetski pasoši za stambene i nestambene zgrade se sastoje od pet strana, dok energetski pasoš za ostale zgrade ima 3 strane.

4.2. Energetski razredi zgrada u zavisnosti od kategorije

Energetski razredi za stambene zgrade određujuse na osnovu maksimalne dozvoljene godišnje potrebne finalne energije za grejanje [kWh/(m^{2}a)], koja je definisana propisom, kojim se uređuju energetska svojstva zgrada na način da su postojeći objekti odvojeni od novih. Maksimalna dozvoljena godišnja potrebna finalna energija za grejanje $Q_{H,nd,max}$ odgovara energetskom razredu „C". Energetski razred zgrade je pokazatelj energetskih svojstava zgrade. Izražen je preko relativne vrednosti godišnje potrošnje finalne energije za grejanje [%] koje se određuje na sledeći način:

$$QH,nd,rel = \frac{QH,nd}{QH,nd,max} x\ 100\%$$

gde je:

$Q_{h,nd,rel}$-relativna vrednost godišnje finalne energije za grejanje [%]

$Q_{h,nd}$- godišnja potrebna energija za grejanje [kWh]

$Q_{h,nd,max}$- maksimalna vrednost godišnje finalne energije za grejanje [%].

4.3. Procedura izdavanja energetskog pasoša

Sistem energetske sertifikacije zgrada uspostavljen je na sledeći način: Ministarstvo nadležno za poslove građevinarstva vodi centralni registar energetskih pasoša i izdaje ovlašćenja organizacijama (privrednim društvima i drugim pravnim licima) za sprovođenje procesa energetske sertifikacije. Energetski pasoš se izdaje nakon obavljenog energetskog pregleda i okončanja finalnog ocenjivanja zahteva vezanih za energetska svojstva zgrade.

4.4. Uloga i značaj energetskog pasoša

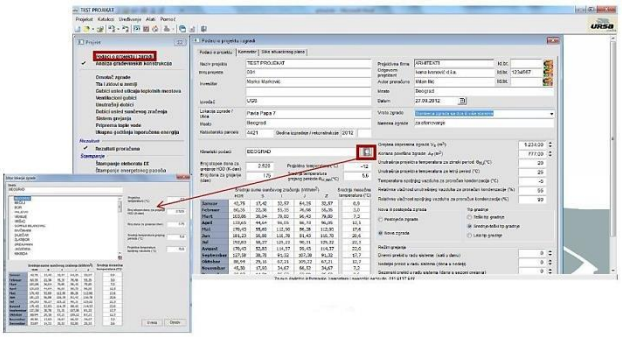
Energetski pasoš za zgrade jeste dokument u kome su predstavljena energetska svojstva zgrade, prema jedinstveno utvrđenoj metodologiji, a služi kao sredstvo informisanja vlasniku zgrade, ministarstvu nadležnom za poslove u oblasti građevinarstva, kao i svim drugim zainteresovanim stranama. Energetska ocena izražava se preko energetskog razreda A+ pa do G, pri čemu je A najefikasnije, a G najneefikasnija kategorija zgrada po pitanju energetske efikasnosti.

5. PROGRAM URSA GRAĐEVINSKA FIZIKA 2

Pri izradi elaborata energetske efikasnosti biće korišćen softver koji je kreirala firma "URSA" d.o.o iz Beograda.

5.1. Kreiranje novog projekta i unošenje osnovnih podataka

Unošenje osnovnih podataka o objektu započinje otvaranjem prozora NOVI PROJEKAT i kreiranje istog.



Slika 6. *Prikaz prozora za unos podataka o projektu*

6. ENERGETSKA SERTIFIKACIJA OBJEKATA

Energetska sertifikacija objekata je izvršena u skladu sa Pravilnikom o uslovima, sadržini i načinu izdavanja sertifikata o energetskim svojstvima zgrada. Predmet energetske sertifikacije jeste 17 objekata. Objekti su različitih karakteristika, funkcionalnosti, kao i površine. Energetski razredi objekata određeni su na osnovu specifične godišnje potrebne toplote u zavisnosti od kategorije zgrade.

Tabela 1. *Parametri objekata preuzeti iz Elaborata EE*

Broj objekata	Klimatski podaci	Neto grejana zapremina zgrade	Specifična godišnja potrebna energija za grejanje	Relativna vrednost godišnje potrošnje finalne energije za grejanje	Ener-getski razred
Objekat1	Novi Sad	109,75	64,83	99,73	C
Objekat 2	Novi Sad	118,45	64,40	99,07	C
Objekat 3	Novi Sad	115,50	62,43	96,04	C
Objekat 4	Novi Sad	137,70	59,14	90,98	C
Objekat 5	Novi Sad	356,85	64,88	99,81	C
Objekat 6	Novi Sad	478,80	64,78	99,66	C
Objekat 7	Šabac	118,60	58,93	90,66	C
Objekat 8	Šabac	422,40	59,27	91,18	C
Objekat 9	Novi Sad	1.082,93	57,18	87,96	C
Objekat 10	Šabac	1.704,85	58,18	34,91	C
Objekat 11	Kraljevo	1.041,36	97,13	58,28	C
Objekat 12	Kraljevo	1.332,20	68,21	40,93	C
Objekat 13	Kragujevac	3.954,60	66,02	39,61	C
Objekat 14	Bor	13.879,90	68,54	41,12	C
Objekat 15	Niš	7.769,45	70,31	42,19	C
Objekat 16	Niš	5.739,03	71,86	43,12	C
Objekat 17	Šabac	7.688,40	71,65	42,99	C

Prilikom energetske sertifikacije objekata relativne vrednosti godišnje finalne energije za grejanje se nalazi u granicama od 34,91% do 99,81%, što zadovoljava kriterijume energetskog razreda C. Prosečna neto grejana zapremina je 2708,86 m^2, prosečna specifična godišnja potrebna energija za grejanje 66,33kWh.

7. UPOREDNA ANALIZA ENERGETSKIH POTREBA OBJEKATA U ZAVISNOSTI OD SISTEMA GREJANJA

U ovom poglavlju izvršiće se uporedna analiza energetskih potreba objekata u zavisnosti od načina grejanja, vrste energenata i načina regulacije sistema grejanja.

7.1. Definisanje parametara za uporednu analizu

Parametar za uporednu analizu jeste potrebna primarna energija E_{prim}. Kategorije su grupisane po površini i funkcionalnosti objekata, a uporedna analiza će biti izvršena u zavisnosti od:

- Načina grejanja (lokalno, centralno, daljinsko);
- Vrsta energenta (prirodni gas, čvrsto gorivo, obnovljivi izvor energije);
- Način regulacije sistema grejanja (lokalna, centralna ili automatska centralna).

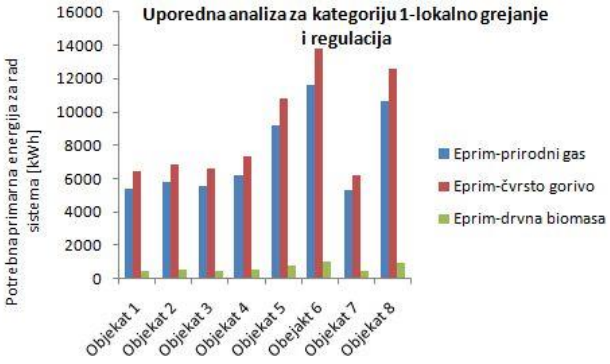
7.2. Rezultati uporedne analize energetskih potreba objekata u zavisnosti od sistema grejanja

U ovom poglavlju biće prikazani rezultati analize u slučaju svih predviđenih kategorija. Izvršiće se uporedna analiza u zavisnosti od slučaja sistema grejanja, načina regulacije sistema i korišćenog energenta.

7.2.1. Rezultati uporedne analize energetskih potreba objekata u zavisnosti od sistema grejanja-kategorija 1

U prvom delu poglavlja izvršiće se poređenje objekata iz kategorije 1, ukoliko je način grejanja lokalni, a energent (prirodni gas, čvrsto gorivo i drvna biomasa) i vrsta regulacije je lokalna. Prekid u sistemu grejanja je 8 sati.

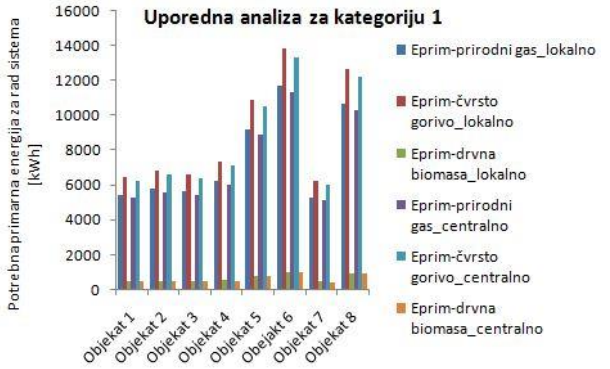
Na slici 7. prikazana je zavisnost energetskih potreba zgrade za kategoriju 1 u zavisnosti od vrste energenata pri lokalnom grejanju i u slučaju lokalne regulacije sistema.



Slika 7. *Energetske potrebe zgrade za kategoriju 1-lokalno grejanje i regulacija*

Prilikom analize u ovom delu korišćen je lokalni sistem grejanja, dok su energenti prirodni gas, čvrsto gorivo-lignit i drvna biomasa. U objektima se nalaze peći koje koriste navedene energente. Toplotni izvori-peći su postavljeni u delovima koje su obuhvaćeni termičkim omotačem. Regulacija se vrši lokalno na samim pećima. Minimalne potrebe za primarnom energijom za rad sistema ima objekat 7, dok najveće potrebe ima objekat 6. Ukoliko je energent prirodni gas, potrebe primarne energije za rad sistema za grejanje suu granicama od 5.400 kWh do 11.709 kWh, dok su potrebe pri korišćenju čvrstog goriva kao energenta u granicama od 6.287 kWh do 13.838 kWh. U slučaju korišćenja drvne biomase potrebe za primarnom energijom na godišnjem nivou su u granicama od 484 kWh do 1.064 kWh. Na osnovu ove analize dolazi se do zaključka da su najmanje potrebe za energijom u slučaju lokalnog grejanja pri korišćenju drvne biomase.

Slika 8. prikazuje zavisnost između potrebne primarne energije za rad sistema kada je lokalno zagrevanje objekta (lokalna regulacija sistema) i kada je centralno zagrevanje (centralna i lokalna regulacija) sa varijacijom energenta.



Slika 8. *Uporedna analiza za kategoriju 1*

Kao zaključak nakon uporedne analize u slučaju objekata iz kategorije 1, može se uočiti da najmanje potrebe za primarnom energijom za rad sistema ima objekat 7 u slučaju kada je centralni sistem grejanja pri korišćenju drvne biomase kao energenta i prilikom centralne i lokalne regulacije. Najveću potrebu za primarnom energijom za rad sistema ima objekat 6, u slučaju kada je

lokalni sistem grejanja i pri korišćenju čvrstog goriva kao energenta prilikom lokalne regulacije sistema grejanja.

8. ZAKLJUČAK

Danas se pojam energetske efikasnost izrazito primenjuje u svim segmentima života. Primena energetske efikasnosti u zgradarstvu u Srbiji nije na adekvatnom nivou. Prosečna potrošnja toplotne energije u Srbiji jeste oko 170 kWh/m^2 dok je ona u zemljama Zapadne Evrope u granicama od 70-130 kWh/m^2. Najveći potencijal raspoloživih mera ili paketa mera za unapređenje energetske efikasnosti jeste upravno u građevinarstvu. Na osnovu člana 201 Zakona o planiranju i izgradnji doneti su pravilnici kojima se propisuju procedure za unapređenje energetske efikasnosti:

- PRAVILNIK O USLOVIMA, SADRŽINI I NAČINU IZDAVANJA SERTIFIKATA O ENERGETSKIM SVOJSTVIMA ZGRADA "Službeni glasnik RS" broj 69/2012;
- PRAVILNIK O ENERGETSKOJ EFIKASNOSTI ZGRADA "Službeni glasnik RS" broj 61/2011.

Na potrebnu primarnu energiju za rad sistema E_{prim} imaju sledeći faktori uticaj:

- ukupan stepen korisnosti postrojenja za grejanje;
- toplotni gubici sistema za grejanje;
- faktor pretvaranja i
- potrebna energija za potrošnju električne energije za sistem grejanja i STV.

Najmanje potrebe za primarnom energijom za rad sistema na godišnjem nivou ima objekat pri korišćenju centralnog grejanja i u slučaju primene drvne biomase kao energenta prilikom centralne regulacije spoljašnje temperature u zavisnosti od uticaja sobne temperature i pri lokalnoj regulaciji na grejnim telima uz pomoć radijatorskih termostatskih ventila.

9. LITERATURA

[1] http://efikasnost.rs/energetska-efikasnost/definicija-energetske-efikasnosti/ preuzeto: 20.07.2019.

[2] http://stanovanje.gov.rs/latinica/energetska-efikasnost.php#prirucnici preuzeto: 24.07.2019.

[3] https://cenazlata.org/energetska-efikasnost-u-zgradama-i-kucama/ preuzeto: 01.08.2019.

[4] Todorović, M., Rajčić, A. *Priručnik za energetsku sertifikaciju (ESZ),* Deutsche Gesellschaft für, Beograd, 2017.

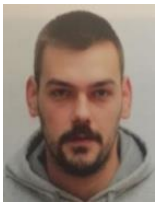
[5] http://stanovanje.gov.rs/doc/energetska-fikasnost/Prirucnici/Prirucnik%20za%20energetsku%20sertifikaciju%20zgrada.pdf preuzeto: 09.08.2019.

[6] Grad Beograd: *Priručnik o energetskoj efikasnosti u stambenim zgradama i kućama*, Beograd, februar 2018.

[7] http://www.pasivnakuca.rs/index.php/pasivna-kuca preuzeto 11.08.2019.

[8] Rakić, G., Kolašinac, F. *Uloga energetske efikasnosti u funkciji očuvanja životne sredine*, 1st International Conference Ecological Safety in Post-modern Environment, Banja Luka, 2009.

Kratka biografija:



Željko Tojaga rođen je u Vojniću 1994. godine. Fakultet tehničkih nauka, odsek Mašinstvo – Energetika i procesna tehnika. Osnovne akademske studije završio je 2017. godine.

kontakt: zeljkotojaga94@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 628.8
DOI: https://doi.org/10.24867/07AM10Sarovic

PRIMENA AUTOMATIZACIJE PODSTANICA U SISTEMIMA DALJINSKOG GREJANJA

APPLICATION OF SUBSTATION AUTOMATION IN DISTRICT HEATING SYSTEMS

Dušan Šarović, *Fakultet tehničkih nauka, Novi Sad*

Oblast- TERMOENERGETIKA

Kratak sadržaj – *Ovaj rad prikazuje kompatibilan izbor opreme za automatizaciju i prilikom ugovaranja radova po sistemu „ključ u ruke“. Ugovaranje, projektovanje, nezaobilazno korišćenje zakonskih okvira, adekvatan izbor i instalaciju automatizovane opreme, za toplotno područje grada Novog Sada, odnosno toplotnih podstanica kod svake grupe potrošača, optimalan izbor i način rada kompletne opreme.*
Ključne reči – *Daljinsko grejanje, automatizacija podstanica*
Abstract - *This paper shows a justified selection of automation equipment when contracting works are on a turnkey basis. Contracting, designing, use of legal frameworks, adequate selection, and installation of automated equipment for the district heating area of the city of Novi Sad. Paper provided a complete solution for thermal substations, optimal choice and operation of its necessary equipment.*
Keywords – *District heating, substation automation*

1. UVOD

Današnjica nalaže racionalizaciju korišćenja, odnosno smanjenje rasipanja energije u celom energetskom lancu: od eksploatacije primarne energije iz prirode, preko transporta, energetskih transformacija i prenosa transformisane energije završno sa korišćenjem finalne energije.

Kako su rezerve fosilnih goriva ograničene, treba ih razumno i racionalno trošiti. Vodeći se tim, potrebe za energijom treba zadovoljiti primenom novih i savremenih energetskih tehnologija koje će najmanje remetiti postojeću ekološku ravnotežu, a sa druge strane uručiti kvalitetan i stabilan vid energije.

Kako je neminovno korišćenje tradicionalnih energenata, intenzivno se radi na usavršavanju tehnologija za eksploataciju i dalju manipulaciju istih, što se jednako odnosi na sve vrste energenata, izmedju ostalih i prirodni gas.

Na sadašnjem stepenu razvoja tehnologija za energetsku transformaciju, zadovoljavanje potreba za toplotnom energijom uz korišćenje prirodnog gasa, kao najčistijeg fosilnog goriva i kao primarne energije u sistemima daljinskog grejanja, na najbolji način zadovoljava racionalizaciju potrošnje u odnosu na ostala fosilna goriva.

Kako postoji tendencija za sve većim korišćenjem toplotne energije u centralizovanim sistemima, potrebno ju je na najbolji način transformisati i plasirati, uz najmanji procenat gubitaka.

Dobra praksa „Novosadske toplane“ na najbolji način pokazuje korišćenje i manipulaciju energijom. Od baznog izvora, prirodnog gasa, transformacije energije kroz kogenerativne sisteme, dalji plasman i distribuciju toplotne energije do krajnjih korisnika kroz regulaciju u toplotnim podstanicama.

Automatizacijom primarnih i sekundarnih toplotnih podstanica kod krajnjih korisnika omogućuje se fina regulacija, plasman toplotne energije ka potrošačima i smanjenje konkretne proizvodnje u izvorima u periodima kada spoljna temperatura raste.

2. USLOVI ZA UGOVARANJE, PROJEKTOVANJE I IZVOĐENJE

Specifikacija postojećeg stanja sa spiskom primarnih podstanica sa pripadajućim sekundarnim toplotnim podstanicama za koje je potrebno dimenzionisati, izraditi projektno rešenje – Projekat za izvođenje -PZI, izabrati i ponuditi mašinsku opremu, pri čemu je predviđena:

Zamena i ugradnja svih hvatača nečistoća na primarnom delu toplotne podstanice ispred regulatora pritiska i na sekundarnom delu toplotne podstanice ispred kombi ventila.

Zamena svih regulatora pritiska koji nisu proizvođača "Danfoss" ili "Samson" – regulatori pritiska moraju biti sa sigurnosnom funcijom (dvostrukom membranom);

Zamena svih sigurnosnih ventila koji nisu proizvođača "Danfoss", "Samson" ili "Berluto";

Ugradnja kombi regulatora protoka sa elektromotornim pogonom na svim regulacionim krugovima za grejanje,

Zamena svih prolaznih ventila na toploj potrošnj vodi - TPV koji nisu proizvođača "Danfoss" ili "Samson", odnosno koji su sa elektromotornim pogonom - EMP koji imaju napon napajanja 24V, strana 8 od 208

Ugradnja senzora temperature i sigurnosnih termostata sa odgovarajućim čaurama od nerđajućeg čelika,

Ugradnja senzora pritiska u 40 referentnih toplotnih podstanica koje će Naručilac definisati pre početka izvođenja radova.

Ugradnja prestrujnog ventila na povratu između hvatača nečistoće i kombi ventila.

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

3. ODABIR ADEKVATNE KLJUČNE OPREME:

Dimenzionisanje i izbor elemenata podstanice pomoću kojih se reguliše radni pritisak, protok primarne vode i obezbeđuje sigurnost od prekoračenja radnog pritiska vrši se na osnovu izračunatih protočnih karakteristika ovih elemanata.

Regulator pritiska je uređaj pomoću koga se reguliše pritisak u sekundarnoj podstanici i kućnoj instalaciji.

Kombi regulator sa elektromotornim pogonom za regulaciju protoka na sekundarnoj podstanici za grejanje.

Prolazni ventil sa elektromotornim pogonom je uređaj za regulaciju primarnoj podstanici za TPV.

U jednoj kućnoj podstanici regulatori pritiska i kombi ventili sa elektromotornim pogonom trebaju biti od istog proizvođača. Za regulaciju pritisaka:

- 2,0 - 5,0 bar
- Δpmin 1bar
- Pritisci otvaranja 4,0 do 6,0 bar
- Izračunati protok za Δp = 0,2 bar
- Nazivni pritisak NP16 bar
- Nazivni otvor min. DN15, max. DN 25
- Pogon sa oprugom Elektromotorni, 230 VAC, IP54, trotačkovni, sa krajnjim prekidačima sa oprugom Elektromotorni, 230 VAC, IP54, trotačkovni, sa krajnjim prekidačima Deklarisanje proizvoda (natpisna pločica na telu ventila i atest).

Ultra zvučno merilo isporučene toplotne energije bazira se na merenju protoka i razlike temperatura nosioca toplote sekundarne podstanice.

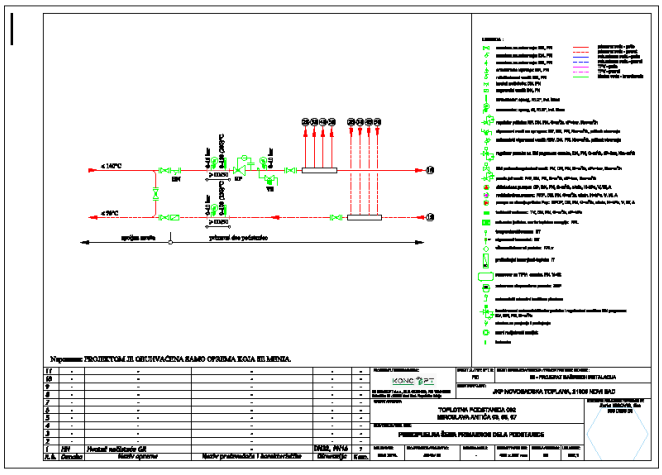
Merilo toplote sastoji se od sledećih elemenata:

1) ultrazvučnog merila protoka,

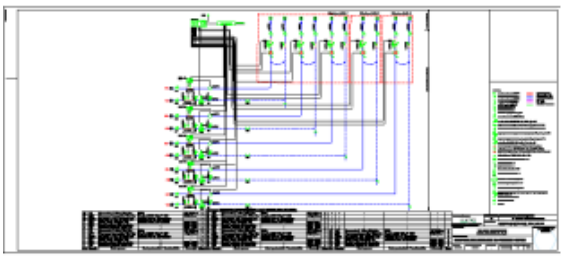
2) t emperaturnih senzor na potis vodu,

3) računske jedinice.

4) imaju odgovarajuće rešenje o odobrenju tipa merila,



Slika 1. *Tehnološka šema primarnog dela automatizovane toplotne podstanice*



Slika 2. *Tehnološka šema primarnog dela automatizovane toplotne podstanice*

3.1. Uređaji za regulaciju pritiska

Slika 3. *Uređaj za regulaciju pritiska*

- Osnovne karakteristike:
- Dimenzije: DN 15-50,
- Maksimalni protok pri padu pritiska od 1 bar do koje ne dolazi do kavitacije unutar uređaja: kVS 4,0-25m^3/h,
- Nazivni pritisak: PN 25,
- Opsezi radnih pritisaka: 1-5/2-8/3-12 bar, Radna temperatura: do 150 °C

3.2. Uređaji za regulaciju pritiska

Slika 4. *Izgled uređaja za održavanje pritiska*

3.3. Uređaji za regulaciju protoka

Slika 5. *Uređaj za regulaciju protoka*

3.3. Uređaji za ultrazvučno merenje potrošnje toplotne energije

Slika 6. *Uređaji za ultrazvučno merenje potrošnje toplotne energije*

Ultrazvučno merilo isporučene toplotne energije bazira se na merenju protoka i razlike temperatura nosioca toplote sekundarne podstanice. Merilo toplote sastoji se od sledećih elemenata:

1) ultrazvučnog merila protoka,

2) temperaturnih senzor na potis vodu,

3) računske jedinice.

4) imaju odgovarajuće rešenje o odobrenju tipa merila,

3.4. Uređaji za merenje, podešavanje i kontrolu radnih parametara

Slika 7. Uređaji za merenje, podešavanje i kontrolu radnih parametara

ECL Comfort 310 je elektronski kontroler za toplotne podstanice u sistemima daljinskog grejanja. Radi na principu regulacije dovodne temperature sekundara zavisno od promene spoljnih uslova. Pripada porodici regulatora ECL Comfort, a koristi se i u sistemima centralnog grejanja i rashladnim sistemima. Ušteda energije može se postići pravilnom regulacijom dovodne temperature u sistemima grejanja i rashladnim sistemima. Moguće je regulisati do 4 kruga.

Funkcija kompenzacije prema spoljnim uslovima u ECL Comfort regulatorima meri spoljnu temperaturu i u skladu sa tim reguliše dovodnu temperaturu za sistem grejanja. Sistem grejanja sa kompenzacijom prema spoljnim uslovima povećava nivo udobnosti i uštedu energije.

ECL Comfort 310 regulator konfiguriše se pomoću izabrane aplikacije putem ECL aplikacionog ključa.

ECL Portal zasnovan na web-u komunicira sa ECL Comfort 310 regulatorom da bi se dobila efikasna i jednostavna SCADA (eng. *Supervisory Control And Data Acquisition*) alatka koja je spremna za korišćenje i predviđena za sve korisnike, servisno osoblje i pri puštanju u rad. Moguće je povećati nivo servisiranja i/ili smanjiti troškove servisiranja. Instalacijama za grejanje i/ili hlađenje može se pristupiti sa gotovo bilo kog mesta u bilo koje vreme preko laptop računara ili pametnih telefona što povećava nivo servisiranja i smanjuje vreme odziva na alarme.

Softver ECL alatke za ECL Comfort 310 nudi mogućnosti alternativne daljinske kontrole u vezi sa ECL Portalom i softverom OPC servera.

ECL Comfort 310 dizajniran je za postizanje komfornih temperatura, optimalnu potrošnju energije, laku instalaciju putem ECL aplikacionog ključa (Plug-and-Play) i jednostavno korišćenje.

Poboljšanu uštedu energije olakšava kompenzacija prema spoljnim uslovima, podešavanje temperature u skladu sa potrebama, optimizacija, kao i ograničenje temperature povrata, protoka i snage.

ECL Comfort 310 lako se koristi putem točkića (multifunkcionalnog tastera) ili jedinice za daljinsko upravljanje (RCU). Točkić i osvetljeni ekran vode korisnika kroz tekstualne menije na izabranom jeziku.

ECL Comfort 310 regulator, između ostalog, ima električne izlaze za regulaciju motornog ventila, relejni izlaz za regulaciju cirkulacione pumpe/ changeover ventila, kao i izlaz alarma.

Moguće je priključiti 6 temperaturnih senzora Pt 1000. Pored toga, 4 ulaza se konfigurišu zavisno od aplikacije. Konfiguracija može biti ulaz temperaturnog senzora Pt 1000, analogni ulaz (0 – 10 V) ili digitalni ulaz.

U zavisnosti od aplikacije, interni dodatni modul ECA 32 (ubacuje se u podnožje regulatora) može davati dodatne ulazne i izlazne signale.

Podnožje je dizajnirano za montiranje na zid ili DIN šinu. Dostupna je varijanta regulatora ECL Comfort 310B (bez

4. ZAKLJUČAK

Cilj ovog rada je bio prikaz racionalnog korišćenja toplotne energije pri distribuciji i generalnom distributivnom sistemu Novosadske toplane. Tendencija sprovođenja definisane strategije i planom automatizacije i povezivanja na sistem daljinskog nadzora i upravljanja

svih primarnih podstanica, što je kao distributer toplotne energije u obavezi prema članu 51 Zakona o efikasnom korišćenju energije („Službeni glasnik RS” 55/2013).

Opštepoznati podatak na području Srbije, jeste da se tokom grejne sezone „baci” velika količina toplotne energije.

Kako dnevni gradijent temperature tokom grejne sezone varira, potreban je sistem koji će odgovoriti zahtevima potrošača a naročito povećati efikasnost transformacije, plasmana i isporuke energije. Upravo se ovakav način kvalitetno isporučene energije postiže automatizacijom prvobitno izvora, zatim distibutivnog sistema i konačno predajne stanice odnosno podstanice. Konkretan primer ugovaranja, projektovanja - odabira adekvatnih elemenata, izvođenja i pratećih pravilnika i zakona opisanih u ovom radu.

Za poboljšanje sistema, adekvatan i dovoljan plasman toplotne energije, Novosadska Toplana je raspisala niz tendera za Automatizaciju toplotnih podstanica. Do današnjeg dana automatizovano je gotovo 70 % toplotnih podstanica, gde je ostvarena velika ušteda.

5. LITERATURA

[1] Proračun toplotne podstanice, Novi Sad, 2015.

[2] Gojnić M, Tehnoekonomska analiza toplotne podstanice, Novi Sad, 2015.

[3] Tehnički opis toplana, Beograd, 2016.

[4] Projektni zadatak Automatizacija podstanica 2018..

[5] Petrović J, SAVREMENE ENERGETSKE TEHNOLOGIJE-, izvodi iz doktorske teze, Novi Sad, 2007

[6] http://www.ingkomora.org.rs/ 27.09.2019.

[7] http://www.aers.rs/ 12.09.2019.

Kratka biografija:

Dušan Šarović rođen je u Novom Sadu 1993. god. Osnovne studije na Fakultetu tehničkih nauka iz oblasti Energetika i procesna tehnika završio je 2016. god. Trenutno student master studija na smeru Energetika i procesna tehnika-termoenergetika.

Зборник радова Факултета техничких наука Нови Сад

УДК: 629

DOI: https://doi.org/10.24867/07AM11Stojkovic

ПОДЕШАВАЊЕ ОСЛАЊАЊА ТАКМИЧАРСКОГ ВОЗИЛА

RACE CAR SETUP

Душан Стојковић, Факултет техничких наука, *Нови Сад*

Област – МАШИНСТВО

Кратак садржај – *Овај рад има за циљ да на основу прикупљених података (возила и возача), и анализе истих, да за предлог подешавања појединачних подсистема у склопу система ослањања возила. Предложена подешавања морају бити у складу са националним правилником категорије „2000 Super Production“.*

Кључне речи: *подешавања ослањања, такмичарско возило, телеметријски подаци*

Abstract – *This paper has a goal to suggest race car setup based on the analysis of used setup and gathered telemetry data. Suggested setups will be in accordance with Serbian National regulations for racing class „2000 Super Production“.*

Keywords: *racing setup, race car, telemetry data*

1. УВОД

Аутомобилизам је спорт у коме такмичар представља кибернетички спој човека и машине. Из наведеног се закључује да успех у овом спорту зависи како од возача, тако и од возила. У свим дисциплинама ауто спорта постоје правила и технички прописи које возач и возило морају да задовоље. Наведена правила имају за циљ да поставе техничка ограничења возилима како би перформансе различитих возила биле приближне, и тиме повећају утицај возача на коначан исход такмичења.

У скоро свим дисциплинама ауто спорта подешавања система ослањања су дозвољена у одређеној или неограниченој мери. Ова подешавања су важна јер одређују релативни положај пнеуматика у односу на подлогу чиме суштински утичу на његово понашање а тиме и на перформансе возила. Утицај подешавања система ослањања се најбоље може видети у резултатима квалификација и трка. Наиме два или више идентичних возила, са различитим подешавањем система ослањања, често остварују потпуно различите резултате и пролазна времена.

2. ПОДАЦИ

2.1 Улазни подаци

Марка возила је „Alfa Romeo 147 Super Production“ које је фабрика наменски израдила за такмичења у категорији „2000 Super Production“.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Борис Стојић, доцент.

Карактеристике мотора:

- Радна запремина: 2000 cm^3
- Максимална снага: 169 kW
- Број обртаја максималне снаге: 7500 $\frac{o}{min}$
- Највећи обртни момент: 200 Nm
- Број обртаја максималног обртног момента: 6500 $\frac{o}{min}$
- Прехрањивање: Атмосверско

Карактеристике возила:

- Погонски точкови: предњи
- Маса возила спремног за вожњу: 1014.5 kg
- Дужина: 4213 mm
- Ширина предњег трага возила: 1801 mm
- Ширина задњег трага возила: 1790 mm
- Међуосовинско растојање: 2546 mm
- Ослањање предњих точкова: двоструко троугаоно раме
- Ослањање задњих точкова: механизам са више водећих полуга *„multi-link“*

Слика 1. *Изглед Возила*

2.1.1 Раподела тежине

Табела 1. Измерене тежине испод сваког точка

Предњи леви точак	Предњи десни точак
3463 N	3419 N
Задњи леви точак	**Задњи десни точак**
2129 N	2036 N

Осовинска оптерећење предње осовине износи: 6882N

Осовинско оптерећење задње осовине износи: 4165N

Помоћу прикупљених података одређује се висина тежишта на следећи начин:

$$H = \frac{L \cdot \sqrt{L^2 - h_D^2}}{h_D} \cdot \frac{R_P' - R_P}{W} + r$$

$$H = \frac{2.546 \cdot \sqrt{2.546^2 \cdot 0.001^2}}{0.001} \cdot \frac{714 - 655}{1014.5} + 0.288$$

$$H = 0.377\ m$$

$R'_P = 714\ kg$ - измерено оптерећење предње осовине када је задњи крај подигнут

$h_D = 0.001\ m$- висина дизања задњег краја

$r = 0.288\ m$- измерени статички радијус точка

2.1.2 Угао бочног нагиба точкова

Од бочног нагиба точка зависи величина контактне површине пнеуматика са подлогом, а самим тим расположива сила приањања приликом убрзања, кочења или скретања возила.

Табела 2. *Измерени бочни нагиби точкова*

Предњи леви точак:	Предњи десни точак:
3.4°	3.7°
Задњи леви точак:	**Задњи десни точак:**
2.1°	2.7°

2.1.3 Угао усмерености точкова

Угао усмерености точкова (енг. „*Toe*“) може бити позитиван, негативан или неутралан. У колико је најмање растојање, у хоризонталној равни, два точка исте осовине испред осовине којима ти точкови припадају тада точкови имају позитиван угао усмерености.

Табела 3. Измерени углови усмерености точокова

Предњи леви точак:	Предњи десни точак:
-0.08°	-0.15°
Задњи леви точак:	**Задњи десни точак:**
0.24°	0.37°

2.1.4 Пнеуматици

Табела 4 Притисци пнеуматика пре вожње

Предњи леви:	Предњи десни:
170 kPa	170 kPa
Задњи леви:	**Задњи десни:**
180 kPa	180 kPa

2.1.5 Телеметријски подаци

За прикупљање телеметријских података, брзине, убрзања, путање, користи се уређај „**LT-Q6000”** произвођача „**Qstarz“.** Прикупљени подаци се затим обрађују у наменском софтверу „**Qstaz QRacing**“.

2.2 Добијени подаци

Одмах по завршетку прве тренинг сесије, извршено је прикупљање података који ће се користити за анализу перформанси возила.

2.2.1 Температуре пнеуматика

Мерење температура пнеуматика извршено је посредством ласерског мерача, на спољашњој, средњој и унутрашњој страни газеће површине.

Табела 5. Измерене температуре пнеуматика

	Предњи леви:			Предњи десни:		
Страна:	Спољашња	Средња	Унутрашња	Унутрашња	Средња	Спољашња
	87 °C	86 °C	84 °C	86 °C	86 °C	84 °C
	Задњи леви:			**Задњи десни:**		
Страна:	Спољашња	Средња	Унутрашња	Унутрашња	Средња	Спољашња
	88 °C	84 °C	82 °C	85 °C	86 °C	85 °C

2.2.2 Притисци пнеуматика

Табела 6. Притисци пнеуматика након вожње

Предњи леви:	Предњи десни:
230 kPa	230 kPa
Задњи леви:	**Задњи десни:**
220 kPa	220 kPa

2.2.3 Телеметријски подаци

Очитавањем са уређаја за прикупљање телеметријских података долази се до вредности брзине и убрзања, као и положаја возила на стази коме ти подаци одговарају, за дати тренутак. Након очитавања података са уређаја, потребно је извршити очитавање вредности максималних бочних убрзања возила за сваку кривину појединачно.

Табела 7. Очитане вредности максималних бочних убрзања у кривинама

Кривина:	Лева (G)	Десна (G)
I	1.40	
II		1.27
III		1.22
IV	1.27	
V	1.50	
VI		1.11
VII	1.53	
VIII		1.20
IX	1.51	

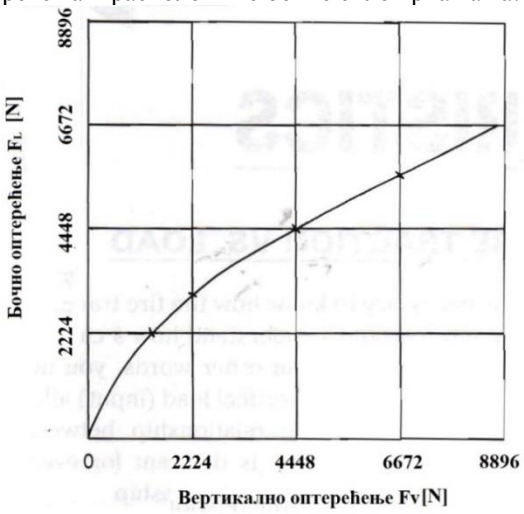
2.2.4 Субјективан утисак возача

При уласку у кривине возач има утисак неутрално подешеног возила по питању подуправљања и надуправљања. На додавању гаса на темену кривине и јачом корекцијом волана ка унутрашњости кривине у малој мери је присутно контролисано надуправљање, које возач сматра пожељним. Приликом јаких кочења возило је стабилно. На стази је примећено доста неравнина које потенцијално могу дестабилизовати возило приликом проласка алтернативним путањама у току трке.

2.3 Обрада података

2.3.1 Пнеуматици

Пнеуматике најбоље карактерише крива вертикалног оптерећења и расположиве бочне силе приањања.



Слика 2. *Крива вертикалног оптерећења и бочне расположиве силе пнеуматика*

Управо је односом максималног могућег бочног оптерећења које се може реализовати при текућем вертикалном оптерећењу, одређена ефикасност скретања [2]. Може се закључити да пораст вертикалног оптерећења пнеуматика негативно утиче на ефикасност скретања јер је пораст бочног оптерећења мањи а самим тим је ефикасност скретања мања.

2.3.1.1 Притисци пнеуматика

На основу измерене вредности притисака пнеуматика (Tabela 8.) и препоруке произвођача истих, закључује се да није потребна корекција притисака.

2.3.1.2 Температуре пнеуматика

На температуре пнеуматика, поред притиска пнеуматика, највећи утицај има геометрија система ослањања возила и расподела динамичких оптерећења на појединачне пнеуматике.

Ако је контактна површина мала и оптерећена приањање је умањено. Поред умањења приањања, долази и до прегревања тог дела пнеуматика, које након одређене вредности изазива трајно умањење силе адхезије. Tabela 7. приказује измерене радне температуре пнеуматика. Из приложеног се може закључити следеће:

- Нагиб предњег левог точка је недовољан.
- Нагиб задњег левог точка је недовољан.
- Нагиб предњег десног точка је превелики.
- Нагиб задњег десног точка је одговарајући.

2.3.2 Бочна убрзања

Из табеле 9 се види да је највеће бочно убрзање левих кривина забележено у седмој кривини и износи 1.53 G. Највеће бочно убрзање десних кривина забележено је у другој кривини и износи 1.27 G.

3. ПРЕДЛОГ ПОДЕШАВАЊА

3.1 Квалификациона подешавања

Током квалификација возач нема потребу претицања других возила на стази (сем ако су доста спорија) тако да је сконцентрисан на своју вожњу, што омогућава примену „агресивнијих“ подешавања која пружају боља пролазна времена али и већи психо-физички напор возача.

3.1.1 Пнеуматици

Предлажи се коришжене вредности притисака пнеуматика као на тренинг сесији.

3.1.2 Спиралне опруге

Возач је приликом тренинг сесије приметио велики број неравнина, тако да се предлажу предње опруге мањих крутости.

Табела 8. *Крутости опруга за квалификације*

Предња лева:	Предња десна:
80 N/mm	80 N/mm
Задња лева:	**Задња десна:**
160 N/mm	160 N/mm

3.1.3 Аморзизери

Због коришћења мекших предњих опруга предлаже се пропорционално смањење карактеристика пригушења предњих амортизера.

Табела 9. *Подешавања амортизера за квалификације*

Напред	
Сабијање:	12
Извлачење:	10
Брзо сабијање:	6
Позади	
Сабијање:	12
Извлачење:	10
Брзо сабијање:	5

3.1.4 Стабилизатори

Како у датом случају возач жели да избегне нагињање возила, а смањена је крутост предњих спиралних опруга, предлаже се повећање крутости предњег стабилизатора. Уз претпоставку да је предњи стабилизатор крући за 5 N/mm врши се провера ефиканости скретања предње осовине.

Како је $E_P = 0.87$ а $E_Z = 0.86$ закључује се да су крутости стабилизатора задовољавајуће.

3.1.5 Геометрија система ослањања

3.1.5.1 На основу прикупљених радних температура пнеуматика (**Error! Reference source not found.Error! Reference source not found.**) и субјективног возачевог осећаја, предлажу се следећи нагиби точкова:

Табела 10 Квалификациони бочни нагиби

Предњи леви точак:	Предњи десни точак:
3,6°	3,6°
Задњи леви точак:	**Задњи десни точак:**
2,7°	2,7°

3.1.5.2 Углови усмерености точкова

Позитиван угао усмерености задњих точкова умањује могућност надуправљања на самом уласку у кривину. Како возачу више погодује надуправљање, позитивна вредност угла усмерености задњих точкова се може умањити. Неутралан угао усмерености умањује отпоре котрљања точкова због чега ће се вредност негативног угла предњих точкова умањити, водећи рачуна да се не компромитује скретање возила.

Табела 11. *Углови усмерености точкова за квалификације*

Предњи леви точак:	Предњи десни точак:
-0.07°	-0.07°
Задњи леви точак:	**Задњи десни точак:**
0.00°	0.00°

3.2 Подешавања за трку

У току трке возач сем своје вожње треба да води рачуна и о вожњи осталих такмичара, и да увек буде спреман на непредвиђен сплет догађаја, брзе реакције на новонастале догађаје као и нагле промене правца кретања.

Из претходно наведеног закључује се да подешавања за трку требају да буду конзервативна тако што ће да обезбеде возачу стабилно возило, ниску деградацију пнеуматика, и добар темпо.

3.2.1 Пнеуматици

Предлажу се исти притисци пнеуматика.

3.2.2 Спиралне опруге

Како су у току трке могуће нагле промене правца, стабилизатори већих крутости би у тим ситуацијама дестабилизовали возило и изазвали губитак контроле. Међутим применом стабилизатора мањих крутости повећава се нагињање возила а самим тим и вертикално оптерећење спољашњих пнеуматика и губитак ефикасности скретања. Да би се нагињање возила умањило биће коришчене спиралне опруге већих крутости.

Табела 12. *Крутости опруга за трку*

Предња лева:	**Предња десна:**
190 N/mm	190 N/mm
Задња лева:	**Задња десна:**
220 N/mm	220 N/mm

3.2.3 Амортизери

Како су за трку предложене опруге већих крутости у односу на тренинг сесију предлаже се пропорционално повећање пригушења сабијања и извлачења.

Табела 13. *Подешавање амортизера за трку*

Напред	
Сабијање:	27
Извлачење:	23
Брзо сабијање:	5
Позади	
Сабијање:	16
Извлачење:	14
Брзо сабијање:	5

3.2.4 Стабилизатори

Прво се извршава прорачунско умањење крутости предњег стабилизатора како би се закључило да ли долази до повећања или смањења расположивог приањања. Ефикасност скретања предње осовине износи $E_P = 0.85$; За трку ће се користити задњи стабилизатор са тренинг сесије док ће задње спиралне опруге бити већих крутости тако да возило тежи благом надуправљању, које погодује скретању. Ефикасност скретања задње осовине износи $E_P = 0.83$ на основу чега се закључује да су крутисти стабилизатора одговарајуће.

3.2.5 Геометрија система ослањања

3.2.5.1 Углови бочних нагиба точкова

Углови бочних нагиба точкова коришћени на квалификавијама пружају одговарајуће температуре пнеуматика због чега ће остати непромењени.

3.2.5.2 Углови усмерености точкова

У току трке проласци кроз кривине могу бити офанзивном, дефанзивном и идеалном путањом. Како би се побољшало скретање возила када се не налази на идеалној путањи, предлаже се негативнија вредност угла усмерености предњих точкова. У случају наглих промена правца кретања возила позитивна вредност угла усмерености задњих точкова умањује могућност настанка превеликог и не контролисаног надуправљања.

Табела 14. *Улгови усмерености точкова за трку*

Предњи леви точак:	**Предњи десни точак:**
-0.10°	-0.10°
Задњи леви точак:	**Задњи десни точак:**
0.03°	0.03°

4. ЗАКЉУЧАК

Извршени прорачуни служе само за основно разумевање понашања возила при различитим подешавањима система ослањања, и пружају смернице ка постизању жељених подешавања. Коришћењем високо прецизних уређаја за прикупљање телеметријских података и поседовање криве преформансе тачно коришћених пнеуматика, ови прорачуни се могу проширити и помоћу њих се доћи до прецизнијих вредности подешавања система ослањања. На тај начин брже би се дошло до жељених подешавања јер би рачунска била приближнија стварно потребним подешавањима. Тако би се ефикасније искористило расположиво време на стази јер би и возач мање времена посветио тестирању подешавања, а више вежбању својих возачких вештина.

5. ЛИТЕРАТУРА

[1] „Appendix-J-Clan-261.“ Спортски Ауто Картинг Савез Србије, .

[2] Herb, Adams. *Chassis Engineering*. H.P. Books, 1993.

[3] http://racingcardynamics.com/racing-tires-lateral-force/. . 25 7 2019.

[4] http://www.alfaclubvic.org.au/forum/index.php?topic=11017.0. . 26 7 2019.

[5] https://alfa-restoration.co.uk/shop/product_info.php?products_id=2007. . 26 7 2019.

[6] https://commons.wikimedia.org/wiki/File:Alfetta_front_suspension_antiroll.jpg. . 5 8 2019.

[7] https://eibach.com/us/p-101-suspension-worksheet.html. . 10 8 2019.

[8] https://en.wikipedia.org/wiki/Anti-roll_bar. . 25 7 2019.

[9] https://www.oponeo.co.uk/blog/camber-understanding-the-wheel-s-angle-of-inclination. . 15 08 2019.

[10] Milliken, William and Douglas Milliken. *Race Car Vehicle Dynamics*. SAE Publications Group, 1995.

[11] Puhn, Fred. *How to make your car handle*. H.P. Books, 1976.

[12] Staniforth, Allan. *Competition Car Suspension*. Haynes Publishing, 1999.

[13] Дубока, Ч. и Ж. Арсенић. *Испитивање моторних возила - Приручник*. Београд: ЈУМВ, 1994.

[14] Стојић, Борис. *Теорија кретања друмских возила*. ФТН Нови Сад, 2014.

Кратка биографија:

Душан Стојковић рођен је у Краљеву 1993. год. Дипломски рад на Факултету техничких наука из области механизације и конструкционог машинства одбранио је 2017. год. након чега је уписао мастер студије.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.7/.9
DOI: https://doi.org/10.24867/07AM12Stanic

OBRADA GLODANJEM PODRŽANA ULTRAZVUKOM

MILLING MACHINING SUPPORTED BY ULTRASONIC

Boban Stanić, Borislav Savković, Nenad Kulundžić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj – *Spoj konvencionalnog glodanja i ultrazvučne obrade, tačnije ultrazvučnih vibracija su rezultovale hibridnoj obradi pod nazivom ultrazvukom podržano glodanje. U ovom radu je pored opisa svake tehnologije, priložen niz eksperimenata u kojima se porede izlazne karakteristike dobijene konvencionalnim glodanjem i ultrazvukom podržanim glodanjem.*

Ključne reči: *Obrada glodanjem, obrada ultrazvukom, ultrazvukom podržano glodanje.*

Abstract – *The combination of conventional milling and ultrasonic machining, more precisely ultrasonic vibrations, resulted in a hybrid machining technology called ultrasonic assisted milling. In addition to the description of each technology, this paper is accompanied by a series of experiments comparing the output characteristics obtained by conventional milling and ultrasonic assisted milling.*

Keywords: *Milling, Ultrasonic Machining, Ultrasonic Assisted Milling.*

1. UVOD

U savremenoj mašinskoj industriji, koja se rapidno razvija, iz dana u dan je sve veća potražnja za kvalitetnijim proizvodima, sa što kraćim vremenom izrade. Pod tim se podrazumeva, pored tačnosti izrade, kvaliteta obrađene površine i drugih izlaznih karakteristika, sam izbor materijala. Savremeni mašinski materijali koji poseduju vrhunske mehaničke osobine, poput Inconel 718, AISI 420 nerđajućeg čelika, staklo-keramike, titanijumovih legura, superlegure na bazi nikla i sl. se masovno koriste vazduhoplovnoj, raketnoj (svemirskoj), automobilskoj i drugim industrijama, pretežno u ekstremnim radnim uslovima, poput povišene temperature, korozione sredine, za odgovorne delove, delove pod konstantnim opterećenjem itd.

Naravno, tako dugovečni i kvalitetni materijali se veoma teško obrađuju, i samim tim predstavljaju ogroman problem konvencionalnim postupcima obrade. Ovaj problem je uslovio inteziviranje hibridne obrade, dakle spoj konvencionalnih postupaka obrade sa nekovencionalnim postupcima. Jedan od mnogih hibridnih postupaka obrade, koji je i bio tema ovog rada, jeste ultrazvukom podržano glodanje. Suština ove obrade jeste ta, da se na alat ili obratak dovede ultrazvučna vibracija, male amplitude, reda veličine μm ili nm, i velike frekvencije, reda veličine kHz.

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

Mnoga eksperimentalna istraživanja, od kojih su neka i izložena, ukazala su na pogodnost i unapređenje izlaznih karakteristika ove obrade u poređenju sa konvencionalnim glodanjem.

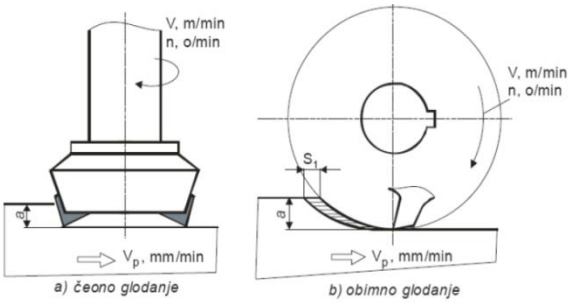
2. OBRADA GLODANJEM

Obrada glodanjem je zbog svoje univerzalnosti i produktivnosti jedna od najzastupljenijih tehnologija obrade rezanjem. Obrada glodanjem je postupak obrade ravnih površina, profilisanih kontura, žlebova, površina složenog i specijalnog oblika [1].

Glavno kretanje je obrtno kretanje alata definisano brzinom rezanja v [m/min]. Pomoćno kretanje je pravolinijsko kretanje predmeta obrade i/ili alata i određeno je brzinom pomoćnog kretanja (v_p = n·s [mm/min] - aksijalnim pomeranjem u jedinici vremena), a može biti definisano korakom po zubu (s_1 [mm/z] - aksijalnim pomeranjem za jedan zub alata) i korakom (s [mm/o] - aksijalnim pomeranjem za jedan obrt alata).

Prema rasporedu reznih elemenata alata razlikuju se dva postupka obrade glodanjem, koja su prikazana na slici 1:

a) čeono glodanje i
b) obimno glodanje.

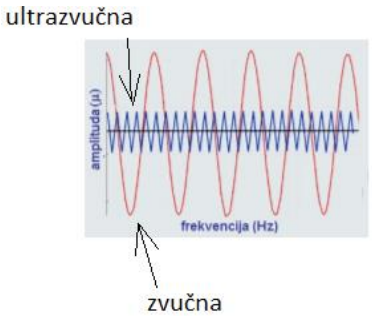


Slika 1. *a) obimno glodanje, b) čeono glodanje* [1]

3. OBRADA ULTRAZVUKOM

Obrada ultrazvukom (*Ultrasonic Machining* -USM) je nekonvencionalni postupak obrade, a svoju primenu je pronašla za dimenzionalnu obradu tvrdih i krtih materijala. Pored toga, ultrazvučna obrada se koristi i za otklanjanje tragova prethodne obrade – poliranje, zaobljavanje oštrih ivica, čišćenje delova i dr. Ultrazvuk se uspešno koristi i za inteziviranje drugih procesa obrade poput: ultrazvukom podržanog glodanja, bušenja, zavarivanja, livenja, plastičnog deformisanja i dr [2].

Ultrazvuk predstavlja elastične talase koji se prostiru određenom brzinom kroz gasovitu, tečnu i čvrstu sredinu, određenom amplitudom oscilacija i talasnom dužinom. Ultrazvuk je zvuk generisan iznad dometa ljudskog sluha, čija frekvencija je obično preko 20 kHz. Na slici 2 je data komparacija frekvencije i amplitude zvuka i ultrazvuka.



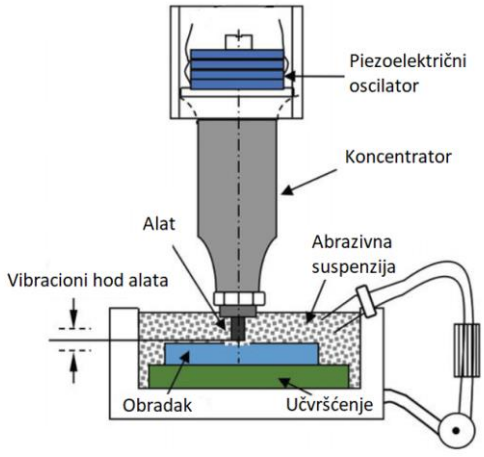
Slika 2. *Frekvencija i amplituda kod ultrazvučne obrade*

Brzina prostiranja zvučnih oscilacija c, zavisi od frekvencije oscilovanja f i dužine zvučnog talasa λ, tj. periode oscilovanja T, izražava se:

$$c = \lambda \cdot f = \frac{\lambda}{T} \quad (1)$$

Pod ultrazvukom se smartraju mehaničke oscilacije visokih frekvencija, koje se dobijaju pomoću elektroakustičnog pretvarača – oscilatora. U oscilatoru se električna energija dobijena od visokofrekventnog generatora pretvara u zvučnu energiju, tj. u mehaničko oscilatorno kretanje. Postoje dve vrste oscilatora, magnetnostrikcijski i piezoelektrični oscilator.

Na slici 3 prikazana je šema osnovnih elemenata USM uređaja. Visokofrekventna električna energija može se pretvoriti u mehaničke vibracije s rezonantnom frekvencijom putem pretvarača. Pobuđena vibracija se potom prenosi kroz pojačivač ultrazvučnih oscilacija – koncentrator, kako bi se pojačala amplituda vibracije i dovela na vrh alata. Abrazivna suspenzija koja sadrži abrazivne čestice u fluidu konstantno se nalazu u zoni obrade. Tokom obrade u radnoj površini dolazi do velikog broja sitnih udara koji dovode do uklanjanja materijala.



Slika 3. *Šema USM uređaja*

4. ULTRAZVUKOM PODRŽANO GLODANJE

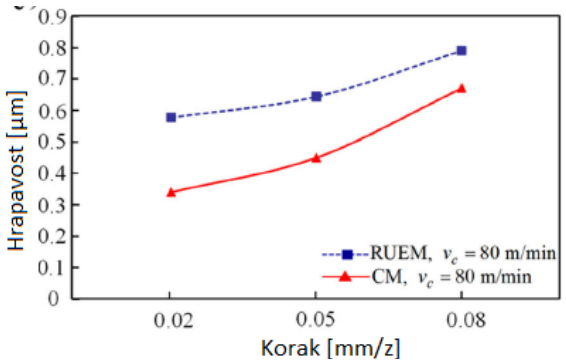
U mašinskoj industriji, tehnologija obrade glodanjem je zbog niza pogodnosti najzastupljeniji proces skidanja materijala. Kako se teži za što tačnijom obradom, potrebno je povećati vrednosti parametara obrade, poput brzine rezanja, pomaka i dubine rezanja. Sa povećanjem vrednosti parametara obrade, generiše se i veća toplotna energija tj. temperatura. Visoka temperatura rezanja nepovoljno utiče na kvalitet obrađene površine kao i na postojanost alata.

Ultrazvukom podržano glodanje (UPG) je napredni proces obrade koji kombinuje ultrazvučne vibracije kao pomoćni mehanizam u procesu glodanja. Razna istraživanja su pokazala da ultrazvukom podržano rezanje doprinosi smanjenju temperature i sile rezanja, zbog manjeg kontaktnog trenja između alata i obratka. Samim tim se postiže veća postojanost, koje može biti čak i do 40% veća, dok se hrapvost obrađene površine može smanjiti i do dva puta. Danas, vibracioni mehanizam se može instalirati direktno na rezni alat ili na radni sto. Vibracija je jedan od mehanizama koji se može koristiti za potpomaganje procesa obrade. Vibracija je oscilacija nekog tela oko njegovog statičkog tj. ravnozežnog položaja. Kada se vibracija male amplitude primeni na rezni alat ili obradak, to pozitivno utiče na proces obrade u poređenju sa itsim, konvencionalnim postukom. Piezoelektrični aktuator je jedan od uređaja koji mogu da proizvesti preciznu amplitudu vibracije (red veličine μm ili nm) visoke frekvencije (red veličine kHz) [3].

Neki od eksperimenata su uključili sledeće materijale: legure titanijuma, staklo-keramika, Inconel 718, AISI 420 nerđajući čelik, legure aluminijuma, stalko, od kojih će nekoliko biti priloženi u nastavku [4-9].

4.1. Uticaj rotaciong ultrazvučnog eliptičnog bočnog glodanja (RUEM) na integritet površine Ti-6Al-4V

Legura titanijuma Ti-6Al-4V je izuzetno atraktivan materijal koji se koristi u vazduhoplovnoj industriji zbog odnosa snage-prema-težini, otpornosti na koroziju i zamor. Međutim, zbog slabe obradivosti, Ti-6Al-4V je podložan površinskom oštećenju i ima malu snagu zamora tokom mašinske obrade. Integritet površine ima važan uticaj na performanse, pouzdanost i trajnost proizvedenih komponenti. U radu [4] eksperiment je izvršen na četvoroosnoj CNC glodalici (BV100) opremljenu držačem alata sa ultrazvučnim eliptoidnim vibracijama. Uticaj brzine rezanja i pomaka na hrapavost obrađene površine prikazan je na slici 4. Sa slike se vidi da se dobija bolja hrapavost obrađane površine tokom konvencionalnog glodanja (CM).

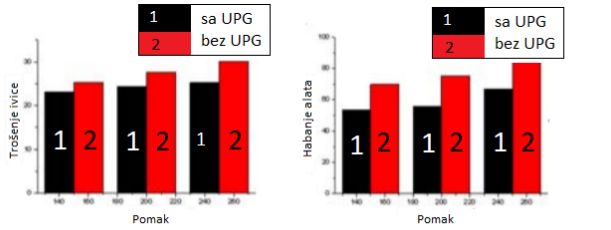


Slika 4. *Uticaj pomaka na hrapavost obrađene površine*

Pored ovog negativnog efekta u pogledu hrapavosti obrađene površine, autori su prikayali i pozitivan doprinos RUEMa. Vrednosti zaostalih pritisnih napona kod rotacionog ultrazvučnog eliptičnog glodanja (*Rotary Ultrasonic Eliptical Milling* – RUEM) mogu se menjati u uskom opsegu variranjem frekvencije vibracija i niži su nego kod CM.

4.2. Uticaj ultrazvukom podržanog glodanja pri obradi stakla-keramike

U procesima rezanja staklo-keramike, visoka temperatura se brzo nakuplja oko rezne ivice alata zbog samog kontakta između obratka i alata kao i zbog vrlo loše toplotne provodljivosti obratka. Slika 5 pokazuje poređenje trošenja ivice i habanja alata nasuprot pomaka sa i bez UPG, pri korišćenju reznog fluida rastovrljivog u vodi [5].

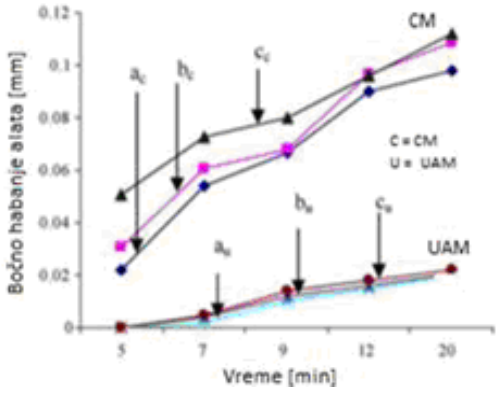


a) Trošenje ivice i pomak b) Habanje alata i pomak

Slika 5. *Poređenje trošenje ivice i habanja alata*

4.3. Izvodljivost ultrazvukom podržanog glodanja za obradu komponenti u vazduhoplovnoj industriji

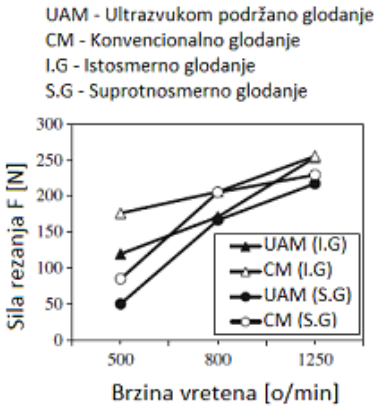
Superlegure su našle svoju primernu za izradu delova namenjenim za rad u ekstremnim uslovima. 80 % upotrebe superlegura kao što su Inconel 718 i titanijum su pronašli u vazduhoplovnoj industriji [6]. Sila rezanja je ključna promenljiva koja se primenjuje kao indikator habanja alata među indirektnim metodama praćenja habanja putem interneta. Rezultati eksperimenta su prikazani na slici 6.



Slika 6. *Habanje alata u zavisnosti od vremena za CM i UAM procese pri pomaku od 0,1 mm/o*

4.4 Uticaj ultrazvučnih vibracija na bočno glodanje AISI 420 nerđajućeg čelika

AISI 420 martenzitni nerđajući čelik se široko koristi na mestima za koje je kombinacija otpornosti na koroziju i velike čvrstoće neophodna, poput lopatica turbina, vratila, propelera i hirurških instrumenata [7]. U ovom eksperimetnu model sile rezanja jednosmernog UAM-a je predstavljen i ultrazvukom podržano glodanje je primenjeno za ispitivanje sile rezanja i kvaliteta obrađene površine u različitim uslovima rezanja. Sa slike 7 se vidi da su sile rezanja kod UAM manje nego kod CM, i to za pomak *afz* = 0.05 mm/z iznosi 19 %.



Slika 7. *Uticaj brzine vretena i smera glodanja na silu rezanja pri pomaku od afz = 0.05 mm/z*

5. EKSPERIMENTALNI RAD

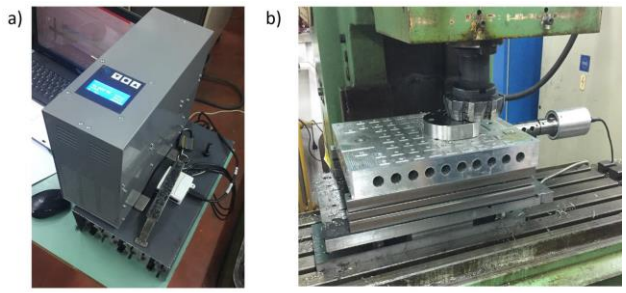
Praktični deo ovog rada prestavlja eksperiment koji je izvršen u laboratoriji za procese obrade rezanjem, smeštene na Departmanu za proizvodno mašinstvo. U ovom eksperimentu je usvojena izvedba UAM-a, gde se vibracija dovodi na radni predmet, leguru titanijuma. Cilj eksperimenta je bio da se uporede izlazne karakteristike, srednja aritmetička hrapavost i maksimalna hrapavost merena u deset tačaka, između konvencionalnog glodanja i ultrazvukom podržanog glodanja.

Eksperiment je izvršen na vertikalnoj glodalici *"PRVOMAJSKA" FSS-GVK-3*. Korišćena glava za čeono glodanje *"JUGOALAT" G.715* ϕ125 mm, sa mehanički pričvršćenim reznim pločicama, sledećih karakteristika: broj zuba $z = 8$, napadni ugao $\kappa = 75°$, grudni ugao $\gamma = 7°$ i leđni ugao $\alpha = 18°$. Kao rezni alat korišćene su rezne pločice kompanije TaeguTec, oznake SPKR 1203 EDR – EM TT8020, presvučene TiCN. Svi eksperimenti su izvođeni sa jednozubim alatom, tj. sa jednom reznom pločicom ($z = 1$). Obradak je legura titanijuma, a njen hemijski sastav i mehaničke osobine su date u Tabeli 1.

Tabela 1. *Hemijski sastav i mehaničke osobine legure titanijuma*

Hemijski element	C	N	H	Al	V	O
Udeo [%]	0.10	0.05	0.015	5.5÷6.75	3.5÷4.5	0.20
Osobina	Zatezna čvrstoća [N/mm2]		Pritisna čvrstoća [N/mm2]		Elongacija 4D	Smanjenje površine
Vrednost	895		828		18	25

Generator koji je korišćen za ovaj eksperiment je proizvođača MPI, snage 2 kW, prikazan na slici 8a). Visokofrekventni generator šalje osilatorsku električnu energiju, koja se pomoću pretvarača i sonotrode pretvara u mehaničku vibraciju i dovodi na obradak. Između radnog stola i obratka se nalazi vibraciona ploča, koja je prvenstveno dizajnirana za rad na erozimatu, slika 8b). Vrednosti su praćene pomoću softvera Ultrasonic Cleaning Generator DSS.



Slika 8. *a) visoko-frekventni generator i b) vibraciona ploča*

Pored vibracione ploče (izrađena od Al7075) u prikazanom sklopu centralno mesto zauzimaju i vibracioni nosači (izrađeni od 316L prohrom) koji imaju za cilj da spreče dalji prenos vibracija ka stolu glodalice.
Vrednost parametara generatora se mogu kontrolisati pomoću softvera Ultrasonic Cleaning Generator DSS, koji je prikazan na slici 9.

Slika 9. *Ultrasonic Cleaning Generator DSS*

Za potrebe ovog istraživanja, tj. merenja hrapavosti obrađene površine, korišćen je uređaj MarSurf PS1, sa mernim rasponom od 350 µm (-200 µm do 150 µm).
U tabeli 2 su dati konstantni parametri eksperimenta. Eksperiment je izvršen bez sredstva za hlađenje i podmazivanje.

Tabela 2. *Parametri eksperimenta*

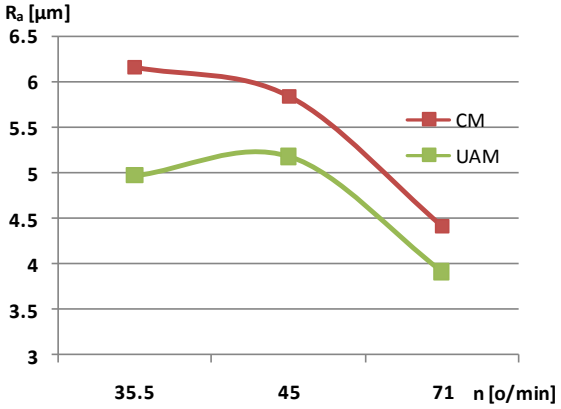
Parametar	Vrednost
Dubina rezanja [mm]	0.5
Pomak [mm/min]	25
Snaga [kW]	1.6
Amplituda [µm]	10÷15
Frekvencija [kHz]	20.85

5.1. Rezultati i diskusija

U tabeli 3 su prikazane dobijene vrednosti hrapavosti. Na slici 10 je prikazana vrednost srednje aritmetičke hrapavosti R_a pri različitim brojevima obrtaja vretena dobijena konvencionalnim glodanjem (CM) i ultrazvukom podržanim glodanjem (UAM).

Tabela 3. *Vrednosti srednje aritmetičke hrapavosti*

Broj obrtaja [o/min]	Srednja aritmetička hrapavost [µm]	
	CM	UAM
35.5	6.157	4.965
45	5.838	5.171
71	4.404	3.906



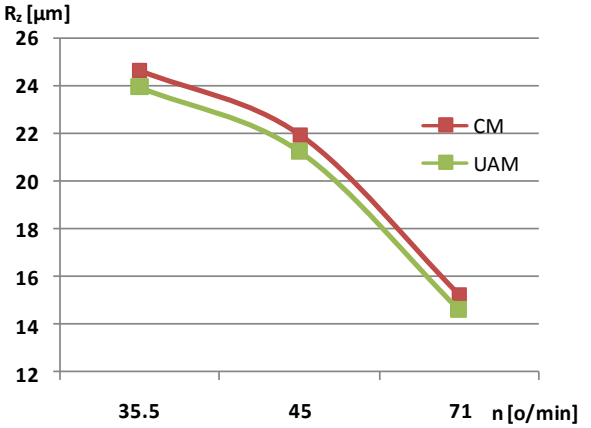
Slika 10. *Srednja aritmetička hrapavost R_a*

Sa slike se jasno vidi da se postižu bolje vrednosti hrapavosti pri UAM-u, a generalno se hrapavost smanjuje sa povećanjem broja obrtaja vretena.

U tabeli 4 su date vrednosti dobijene maksimalne hrapavosti merene u deset tačaka. Slika 11 prikazuje vrednost maksimalne hrapavosti merene u deset tačaka R_z pri različitim brojevima obrtaja vretena, dobijena konvencionalnim glodanjem (CM) i ultrazvukom podržanim glodanjem (UAM).

Tabela 4. *Vrednosti maksimalne hrapavosti merene u deset tačaka*

Broj obrtaja [o/min]	Maksimalna hrapavost merena u deset tačaka [µm]	
	CM	UAM
35.5	24.6	23.9
45	21.9	21.2
71	15.2	14.6



Slika 11. *Maksimalna hrapavost merena u deset tačaka R_z*

Sa slika 10 i 11 vidi se da je postignuta manja hrapavost kod UAM obrade u odnosu na CM. U eksperimentu su varirane samo minimalne brzine rezanja iz razloga sto pri višim brzinama dolazi do oštećenja rezne pločice usled povišenih temperatura rezanja. Može se zaključiti da eksperimentalna izvedba donosi nižu hrapavost pri obradi navedene legure titanijuma, što nije bio slučaj u radu [4].

6. ZAKLJUČAK

Sa razvojem mašinske industrije, rastu i apetiti kupaca. Ne samo u pogledu potrebnog vremena za izradu željenog proizvoda i kvaliteta izrade, već i pri izboru materijala. Novi materijali se pojavljuju na tržištu sa sve boljim mehaničkim osobinama, što otežava njihovu obradivost. Teško obradivi materijali zahtevaju pojačane radne uslove koji se stavljaju pred mašinu, i to u vidu povećanih sila rezanja, pomaka i brzina rezanja. Povećanje radnih uslova rezultira povećanoj radnoj temperaturi što veoma nepogodno utiče na izlazne karakteristike obrade. Stoga, teži se ka inteziviranju hibridnih procesa obrade.

Obrada savremenih materijala konvencionalnim glodanjem je moguća, ali po cenu dužeg vremena izrade, intezivnog habanja alata, smanjene tačnosti i kvaliteta obrađene površine, povećanih sila rezanja i još mnogo faktora. Da bi se ovi faktori smanjili, došlo je do spoja dva, konvencionalna i nekonvencionalna postupka obrade, glodanje i ultrazvučna obrada, u jednu funkcionalnu celinu nazvanu ultrazvukom podržano glodanje (Ultrasonic Assisted Milling – UAM).

Neki od savremenih materijala koji su bili obrađeni ultrazvukom podržanim glodanjem su Inconel 718, AISI 420 nerđajući čelik, legure titanijuma, superlegure na bazi nikla, staklo-keramika i dr. su dali jasne rezultate kako konstantan dovod vibracije može pozitivno uticati na njihovu obradivost i same izlazne karakteristike. Razlog tome je što vibracija utiče na učestalije odvajanje kontaktne površine alat – obradak, a samim tim dolazi do boljeg protoka sredstva za hlađenje i podmazivanje i odvođenja akumulirane radne toplote.

Može se zaključiti da je glodanje podržano ultrazvukom dobra tehnologija obrade i da se najbolji rezultati postižu u eksperimentima u kojima je predmet obrade bilo staklo. Iz ovoga se može zaključiti da je UAM najpogodniji za obradu tvrdih i krtih materijala.

7. LITERATURA

[1] D. Milikić, M. Gostimirović, M. Sekulić: *Osnove tehnologije obrade rezanjem*, Fakultet tehničkih nauka, Novi Sad, 2008.

[2] M. Gostimirović: *Nekonvencionalni postupi obrade*, Fakultet tehničkih nauka, Novi Sad, 2009.

[3] A. Latif: *Experimental study of biomedical stainless steel 316L vibration assisted milling using retrofittable 1D UVAM worktable*, Faculty of Mechanical and Manufacturing Engineering, Johor, 2017.

[4] J. Liu, X. Jiang, X. Han, Z. Gao, D. Zhang: *Effects of rotary ultrasonic elliptical machining for side milling on the surface integrity of Ti-6Al-4V*, School of Mechanical Engineering and Automation, Beijing, 2018.

[5] S. Lin, C. Kuan, C. She, W. Wang: *Application of Ultrasonic Assisted Machining Technique for Glass-Ceramic Milling*, World Academy of Science, Engineering and Technology International Journal of Mechanical and Mechatronics Engineering Vol:9, No:5, 2015.

[6] M. Hafiz, M. Kawaz, W. Mohamad, M. Kasim, R. Izamshah, J. Saedon, S. Mohamed: *A review on feasibility study of ultrasonic assisted machining on aircraft component manufacturing*, Faculty of Manufacturing Engineering, Malaysia, 2017.

[7] M. M. Abootorabi Zarchi, M. R. Razfar, A. Abdullah: *Influence of ultrasonic vibrations on side milling of AISI 420 stainless steel*, Faculty of Mechanical Engineering, Teheran, 2012.

[8] J. Zhang, H. Li, J. Wang, X. Wang, X. Shen: *Ultrasonic vibration-assisted milling of aluminum alloy*, Department of Mechanical Engineering, Jinan, 2012.

[9] J. Xiaoliang, X. Bouyan: *Experimental study on surface generation in vibration-assisted micro-milling of glass*, Oklahoma State University, Stillwater, 2015.

Kratka biografija:

Boban Stanić rođen je u Vrbasu 1994. god. Diplomirao je na Fakultetu tehničkih nauka, smer Proizvodno mašinstvo 2017. god. Master studije upisao je iste godine na usmerenju CIM.

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

Nenad Kulundžić rođen je u Kraljevu 1988. god. Diplomski-master rad na Fakultetu tehničkih nauka iz oblasti Proizvodnog mašinstva iz predmeta Visokoproduktivne obrade odbranio je 2013.god. Od 2018. je u zvanju istraživača saradnika.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK 532.51:519.87
DOI: https://doi.org/10.24867/07AM13Ratkovac

RAZVOJ ZAPTIVNIH PROFILA PRIMENOM RAČUNARSKE DINAMIKE FLUIDA

DEVELOPMENT OF SEALING PROFILES USING COMPUTATIONAL FLUID DYNAMICS

Ivana Ratkovac, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MAŠINSTVO

Kratak sadržaj – *U ovom radu prikazan je razvoj zaptivnih profila korišćenjem računarske dinamike fluida. Opisan je razvoj ekstruzione matrice i njena proba na ekstruzionoj liniji. Dobijen je poprečni presek profila, ti uzorci su testirani u laboratoriji. Na osnovu laboratorijskih rezultata utvrđeno je na kom nivou razvijenosti je taj profil. Takođe, utvrđeno je u kom smeru dalje treba da ide razvoj profila.*

Abstract – *The development of sealing profiles using computational fluid dynamics is presented in this paper. The development of the extrusion matrix and its trial on the extrusion line are described. A profile cross section was obtained. These samples were tested in the laboratory. Based on laboratory results, it was determined the profile development level. Also, it was determined in which direction the profile development should go*.

Ključne reči: *Zaptivni profili, računarska dinamika fluida, ekstruzija.*

1. UVOD

Od kako je pronađen postupak vulkanizacije, guma je zauzela izuzetno mesto među materijalima koji karakterišu savremenu civilizaciju. Iako obično nije osnovni konstrukcioni materijal, zbog svojih elastičnih, izolacionih i zaptivnih svojstava, upotreba gume se toliko proširila, da bi njen iznenadni nestanak izazvao zastoj i poremećaje u savremenoj civilizaciji. Za proizvodnju zaptivnih profila u automobilskoj industriji i građevinarstvu najčešće se koriste elastomeri na bazi sintetičkog etilen propilen dienskog kaučuka (EPDM). Za razvoj zaptivnih profila u automobilskoj industriji koristi se računarska dinamika fluida i metod pokušaja i proba.

2. ZADATAK RADA

Zadatak rada je da se korišćenjem računarske dinamike fluida razvije ekstruziona matrica i da se sa tom matricom uradi probna ekstruzija, što predstavlja metod pokušaja i proba. Ekstrudirani profil mora da zadovolji funkcionalne i vizuelne karakteristike date tehničkim crtežom. Na osnovu dobijenih laboratorijskih rezultata potrebno je predložiti dalji tok razvoja profila.

NAPOMENA:

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

3. OPŠTE O KAUČUKU, GUMI I ZAPTIVNIM PROFILIMA

Pod kaučukom se podrazumeva neumrežen i makromolekulski material prirodnog ili sintetičkog porekla, od koga se proizvodi guma. Guma je makromolekulski materijal, dobijen od kaučuka, koji ima svojstva:

- da se na sobnoj temperaturi brzo vraća u prvobitni oblik;
- da ne može da se lako preoblikuje u stalan oblik.

U zavisnosti od strukture polaznih sirovina proističu specifična svojstva ovih materijala koja su našla primenu u različitim granama industrije. U sastav smeše za dobijanje elastomernog materijala, pored kaučuka i umrežavajućeg sistema, umešavaju se i razni dodaci (aditivi) u cilju poboljšanja fizičko-mehaničkih svojstava, smanjenja cene koštanja i postizanja specifičnih svojstava konačnih materijala. Svojstva elastomernih materijala mogu se podeliti u dve najvažnije grupe: fizička i mehanička.

Za kaučuke je od posebnog značaja viskoelastično stanje, to jest fizičko stanj ekaučuka, koje pri sobnoj temperaturi, karakteriše visoka viskoznost i u isto vreme, velika pokretljivost pojedinih segmenata makromolekularnog lanca. U tom stanju kaučuci poseduju još jedno od važnih fizičkih svojstava - veliku elastičnost, tj. sposobnost da se povratno deformišu pod dejstvom relativno malih opterećenja. Mahanička svojstva gume mogu se određivati u statičkim uslovima, tj. pri stalnim opterećenjima i deformacijama, pri malim brzinama delovanja opterećenja, a takođe i u dinamičkim uslovima, na primer, pri višekratnim deformacijama istezanja, sabijanja, savijanja.

Pri tome kod gume se često ispituje izdržljivost na zamor i obrazovanje toplote pri sabijanju. Elastomerni materijali poseduju dobra zaptivna svojstva jer se odlikuju malom tvrdoćom, a nepropustljivi su i inertni za mnoge fluide. Postizanje pouzdanog zaptivanja zazora, a da se pri tome ne naruši funkcionalnost tog spoja, složen je zadatak. Jedna od metoda je kontaktna metoda koja podrazumeva ubacivanje pomoćnog, mekšeg materijala između dodirnih površina koji ima sposobnost da popuni zazor.

Što je materijal elastičniji, to on bolje popunjava zazor i bolje zaptiva. Istovremeno, takav materijal ne sme izlaziti iz zazora pod dejstvom površinskog pritiska i ne sme se oštetiti usled mehaničkih uticaja. Pored toga, zaptivka mora da prati sve promene zazora, slučajne ili funkcionalne. Zbog toga, materijal, pored elastičnosti mora imati i visoku mehaničku čvrstoću i dobru reverzibilnu elastičnost.

Vulkanizovana guma je upravo jedan od takvih materijala, koji ima sposobnost da značajno izmeni svoj oblik pod dejstvom pritiska. Zbog tog svojstva gume ona je jedan od osnovnih zaptivnih materijala za sve vrste spojeva.

4. UVOD U RAČUNARSKU DINAMIKU FLUIDA

U računarskoj dinamici fluida koriste se parcijalne diferencijalne jednačine zakona održanja mase, promene količine kretanja i energije. Računarska dinamika fluida može kvalitativno predvideti protok fluida. Po pravilu računarska dinamika fluida ne zamenjuje merenja u potpunosti, ali se broj eksperimenata i ukupni troškovi primenom računarske dinamike fluida mogu značajno smanjiti [1].

Rezultati simulacije nikad nisu 100% pouzdani jer:

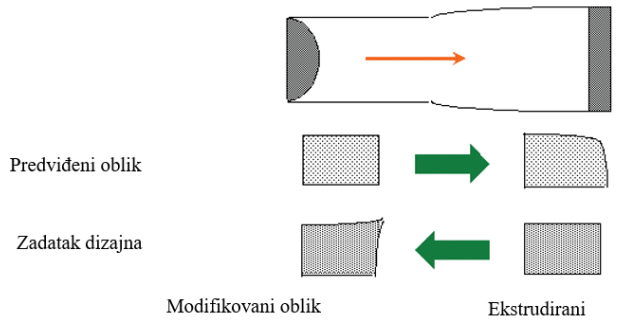
- ulazni podaci mogu biti neprecizni;
- matematički model može biti neadekvatan i
- tačnost rezultata je ograničena raspoloživom računarskom snagom.

Ekstruzija je proces koji se koristi za proizvodnju profila fiksnog poprečnog preseka. Materijal pod pritiskom prolazi kroz matricu željenog poprečnog preseka. Dve glavne prednosti ovog procesa u odnosu na ostale proizvodne procese je mogućnost proizvodnje vrlo složenih poprečnih preseka i obrada krhkih materijala, jer na materijal utiču samo sile pritiska i smicanja.

Ekstruzija je jedan od najčešće korišćenih procesa za obradu polimera. Primeri ekstruzije polimera obuhvataju: folije, cevi, ploče, profile, žice, kablove i koekstruziju premaza. U svakoj vrsti ekstruzije, polimer pod pritiskom prolazi kroz matricu za dobijanje željenog oblika. Da bi se zadovoljio trenutni i predstojeći rast potražnje kompleksnih ekstrudiranih oblika potreban je precizan i brz dizajn od samog početka. Dobijanje konačnog oblika ekstrudiranog proizvoda je proces koji traje i prouzrokuje mnogo grešaka. Nekad je potrebno deset i više proba da bi se dobio proizvod po specifikaciji. Za iskusnog projektanta koji radi na jednom obliku, ovaj proces može biti kraći, a za složenije oblike može trajati nekoliko meseci koristeći metod probe i graške. Postoji mnogo razloga za ovo kašnjnje, od kojih većina uključuje nedostatak razumevanja fizike i protoka polimera unutar i izvan matrice. Projektanti znaju zahtevani oblik profila ali ne znaju odgovarajući oblik matrice.

Vrši se predviđanje oblika ekstrudiranog profila u odnosu na dizajn matrice. Prilagođavanjem oblika matrice, projektant dobija odgovarajući oblik profila i upoređuje ga sa zahtevanim; ako se ne poklapa proces se ponavlja dok se ne dobije zahtevani oblik. U suštini, ovo je metod probe i grešake koji se oslanja na iskustvo projektanta. Predviđanje oblika ekstrudiranog profila takođe se može postići korišćenjem numeričke simulacije.

Korišćenjem računarske simulacije, projektant izračunava deformacije slobodnih površina koje izlaze iz matrice. Do deformacije slobodnih površina može doći zbog preusmeravanja brzine usled iznenadnog odsustva trenja izvan matrice ili zbog smanjenog pritiska profila zbog materijala sa značajnom viskoelastičnom komponentom. Kao rezultat ovih predviđenih deformacija, neophodno je projektovati matricu koja odgovara ovim promenama oblika kako bi se dobio zahtevani oblik profila.



Slika 4.1. *Izgled matrice i površine slobodnog protoka* [2]

Razumevanje šta može da izazove promene oblika profila je ključno za poboljšanje procesa projektovanja - dizajniranja. Na primer, unutar matrice, primećem je paraboličan oblik brzine sa nultom i malom brzinom blizu zida, zbog trenja duž zida koje usporava protok. U centru sekcije toka, brzina smeše je mnogo veća (gornji deo *slike 4.1*). U slobodnom protoku neposredno posle matrice, profil brzine je ravnomeran, svaka čestica ima istu brzinu, jer nema zida koji ih usporava. S obzirom na ove uslove, smeša koja teče u blizini zida unutar matrice moraće da se ubrza do prosečne vrednosti brzine u slobodnom mlazu. S obzirom na konstantan lokalni protok, jedini način za ubrzanje smeše je smanjenje protoka. S druge strane, smeša koja teče u centru kanala matrice ima veću brzinu, taj deo mora da se uspori. Imajući u vidu konstantan lokalni protok, jasno je zašto dolazi do nekontrolisanog rasta profila na izlazu iz matrice. Redistribucija brzine javlja se i kada su uključeni trodimenzionalni efekti, ali osnovni principi ostaju isti.

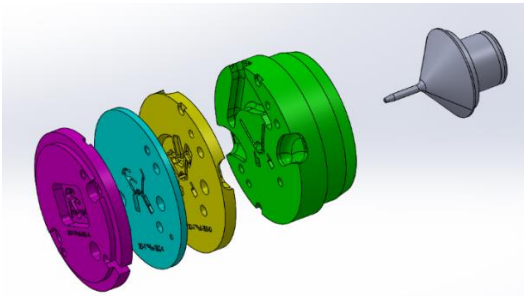
5. PROCES RAZVOJA PROFILA

Na osnovu specifikacija definisanih tehničkim crtežom počinje proces razvoja profila koji treba da zadovolji sve zahteve, a to su:

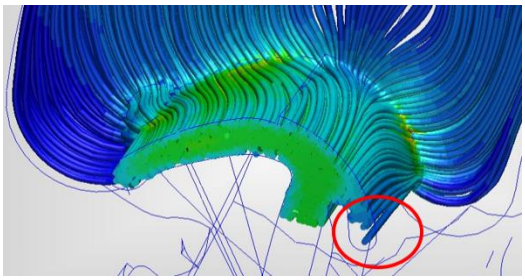
- materijali;
- brzina;
- oblik;
- dimenzije;
- funkcionalne karakteristike i
- vizuelne karakteristike.

Koraci pri razvoju matrice:

1. analiza profila prema crtežu;
2. prilagođavanje profila tipskoj tehnologiji;
3. dodavanje stabilizacionih elemenata na profil;
4. utvrđivanje konačne izlazne konture;
5. utvrđivanje rasporeda ekstrudera;
6. određivanje površina poprečnih preseka;
7. određivanje zapreminskog protoka i
8. konstruisanje ekstruzione matrice (*slika 5.1*).
9. simulacija protoka kroz ekstruzionu matricu, *slika 5.2*;
10. izmena geometrije ekstruzione matrice se vrši na osnovu rezultata simulacije ili rezultata kontrole uzoraka profila sa probne proizvodnje i
11. proba proizvodnje.



Slika 5.1 Delovi matrice



Slika 5.2 Kritična tačka profila brzine

Koraci 9, 10 i 11 predstavljaju petlju iteracionog procesa razvoja ekstruzione matrice. Drugim rečima, ponavljanjem ova tri koraka po njihovom navedenom redosledu dolazi se do karakteristika proizvoda koje su propisane crtežom.

Urađena je proba fizički na ekstruzionoj liniji. Cilj probe bio je da se vidi nivo razvijenosti profila i da se urade laboratorijske analize svih propisanih zahteva. Dobijen je poprečni presek profila prikazan na *slikama 5.3 i 5.4*.

Slika 5.3 Poprečni presek profila

Slika 5.4 Kritični deo profila

6. LABORATORIJSKO TESTIRANJE

Pri procesu razvoja profila, najbitnije je znati koje zahteve je profil ispunio a koje nije, da bi se znalo u kom smeru treba da ide razvoj, i koje su kritične tačke. Jedini način da se ti rezultati dobiju, jeste da se uradi detaljna laboratorijska analiza.

Svi funkcionalni testovi (kontaktni pritisak usana, montiranje i demontiranje) su dobri. Testovi materijala poput tvrdoće i abrazije floka nisu dobri. Nakon dobijenih rezultata laboratorijskog testiranja, dobija se jasna slika na kom je stepenu razvijenosti profil i u kom smeru treba da ide dalji razvoj.

Kritični deo profila za koji je simulacija pokazala da neće imati dovoljan protok materijala, dobijen je u minimalnoj dimenziji povećavanjem broja obrtaja (RPM) do maksimuma, ali mu položaj nije dobar. Ali ipak je ispunjen zahtev testa za kontaktni pritisak usana.

7. ZAKLJUČAK

Predmet ovog rada jeste razvoj zaptivnog profila u automobilskoj industriji korišćenjem računarske dinamike fluida i metode probe i greške.

Pošto se metoda probe i greške bazira na iskustvu projektanta, cilj je bio da se pomoću numeričke simulacije detektuju sve potencijalne nepravilnosti na ekstruzionoj matrici, koje bi prouzrokovala loše karakteristike zaptivnog profila. Rezultati računarske simulacije mogu pokazati i potencijalne nepravilnosti, koje kad se unapred detektuju mogu rešiti promenom parametara procesa.

Nakon odrađene računarske simulacije i uviđanja potencijalnih nepravilnosti, sa tom matricom urađena je proba na ekstruzionoj liniji. Kao što je simulacija i pokazala, pojavile su se nepravilnosti na profilu, koje su podešavanjem procesnih parametara korigovane da bi ispunile tolerancije propisane crtežom.

Kolektovana je dovoljna količina profila da bi se uradili laboratorijski testovi, jer je to jedini način kojim se može utvrditi da li profil ispunjava sve karakteristike propisane crtežom.

Laboratorijski rezultati su pokazali da su svi funkcionalni testovi dobri, a da testovi materijala (tvrdoća i abrazija floka) nisu dobri. Cilj je postignut, korišćenjem računarske simulacije dobijen je profil zadovoljavajućeg poprečnog preseka.

Rezultati su pokazali da je prioritet podešavanje parametara procesa, i tek nakon toga korekcija dizajna matrice u cilju da poprečni presek profila zadovoljava nominalnu vrednost sa crteža.

Potrebno je da se uradi još jedna proba na ekstruzionoj liniji sa istom matricom, da bi se podesili parametri u odnosu na laboratorijske testove materijala koji nisu dobri. Nakon toga potrebno je ponovo uraditi laboratorijsko testiranje profila koji je ekstrudiran sa izmenjenim procesnim parametrima. U odnosu na dobijene rezultate nastavlja se razvoj profila korišćenjem računarske simulacije i metode probe i greške.

8. LITERATURA

[1] Horvat Zvonimir: Tehnologija gume, Udruženje preduzeća za industriju gume FHRJ, Beograd, 1960.

[2] Hossam Metwally: A Methodology for superior Die design – Combining the best of art and science. www.ansys.com,http://www.ozeninc.com/wp-content/uploads/2015/07/Ozen-Engineering-_A-Methodology-for-Superior-Die-Design-Combining-the-Best-of-Art-and-Science-White-Paper.pdf pristupljeno 22.10.2019

Kratka biografija:

Ivana Ratkovac rođena je u Derventi, BiH 1991. god. Diplomski-master rad na Fakultetu tehničkih nauka iz oblasti Mašinstva – Gasna i naftna tehnika odbranila je 2015.god.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.397.13
DOI: https://doi.org/10.24867/07BE00Jankovic

UPOREDNA ANALIZA STANDARDA H.264 I H.265

COMPARATIVE ANALYSIS OF H.264 AND H.265 STANDARDS

Mara Janković, Željen Trpovski, Dejan Nemec, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu prikazane su pojedine tehnike primenjene u standardu H.265 (HEVC). Data je analiza ovih tehnika u poređenju sa odgovarajućim tehnikama iz standarda H.264.*

Ključne reči: *HEVC, H.264, H.265, Quadtree, Intra Predikcija, Inter Predikcija*

Abstract – *In this paper techniques used in the standard H.265 are displayed. Analysis of these techniques is given in comparison with the corresponding techniques in the standard H.264.*

Keywords: *HEVC, H.264, H.265, Quadtree, Intra Prediction, Inter Prediction*

1. UVOD

Televiziju mnogi smatraju za najznačajniji izum u istoriji komunikacije posle pisane reči. Prednost televizije u odnosu na pisanu reč jeste ta što za nju ne postoje nepismeni. Zato ona može dopreti do većeg broja ljudi.

Razvoj digitalne televizije započeo je u poslednjoj dekadi 20. veka. Digitalizacija TV signala veoma je komplikovan proces sa puno složenih detalja i tehnika. Standard H.265 (HEVC – *High Efficiency Video Coding*) jeste najnoviji standard u razvoju digitalne televizije. Predviđeno je da nasledi H.264/AVC (*Advanced Video Coding*).

Glavni razlog za razvoj HEVC-a jeste povećana potreba korisnika za video sadržajima visoke rezolucije 4k/8k ultra HD (*High Definition*). Potrebno je i preneti video sadržaj ovih rezolucija putem interneta i prikazati ga na zahtev. Istu uslugu potrebno je pružiti i korisnicima mobilnih uređaja uz prihvatljivu potrošnju baterije tih uređaja. Povećava se ponuda videa ovakve rezolucije. Medijska kuća "Netflix" počela je 2014. god. da strimuje sadržaj 4k rezolucije. "Amazon" takođe nudi sadržaj ove rezolucije.

Video servis "Youtube" dozvoljava postavljanje videa visoke rezolucije. Mnoge televizijske kuće pružaju mogućnost gledanja video sadržaja 4k/8k rezolucije. U našoj zemlji jedan kablovski operater nudi praćenje sportskih sadržaja u rezoluciji 4k. Naravno, potrebne su i kamere za snimanje videa ovih rezolucija. Uvođenjem visokih rezolucija potrebna je efikasnija kompresija podataka.

U ovom radu dat je pregled pojedinih tehnika H.265 standarda, uporedno u odnosu na H.264 standard.

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

2. SINTAKSA I SEMANTIKA

U standardu H.265 definisana je sintaksa koja pojednostavljuje implementaciju i povećava kompatibilnost. Kao i prethodnici, H.265 propisuje standarde za protok bita, standarde za sintaksu i daje pravila pomoću kojih će svaki dekoder, kada na ulazu stigne signal koji je u skladu sa standardom, na izlazu dati identičnu sliku.

2.1 Set parametara

Standard H.265, kao i H.264, koristi „*high-level*" sintaksnu arhitekturu. Iz H.264 preuzeti su setovi: sekvenca parametra seta (SPS – *Sequence Parameter Set*) i set parametra slike (PPS – *Picture Parameter Set*). U H.265 uveden je novi set, set video parametra (VPS – *Video Parameter Set*), koji daje informacije o profilima i maksimalnim nivoima. VPS opisuje osobine kodovanih video sekvenci, zavisnost između vremenskih podslojeva ili višestrukih 3D prikaza. Ovo predstavlja unapređenje u odnosu na H.264 višeslojne ili „*multiview*" ekstenzije (koriste SEI (*Supplemental Enhancement Information*) poruke da bi prenele ovakve informacije).

U sloju video kodovanja (VCL – *Video Coding Layer*) podržani su svi niski nivoi procesiranja signala. To su particionisanje bloka, intra i inter predikcija, transformacijsko i entropijsko kodovanje i filtriranje („*loop filtering*").

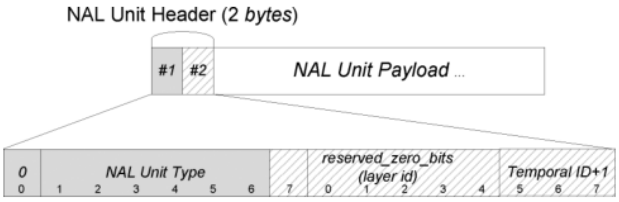
U sloju mreže apstrakcije (NAL) vrši se inkapsuliranje kodovanih podataka i odgovarajućih informacija signalizacije u NAL (*Network Abstraction Layer*) jedinice. NAL jedinice su pogodne za video transmisiju. NAL mapira podatke dobijene iz VCL-a. U tabeli 1. dati su tipovi NAL jedninica.

Tabela 1. *Tipovi NAL jedinica [1]*

Type	Meaning	Class
0, 1	Slice segment of ordinary trailing picture	VCL
2, 3	Slice segment of TSA picture	VCL
4, 5	Slice segment of STSA picture	VCL
6, 7	Slice segment of RADL picture	VCL
8, 9	Slice segment of RASL picture	VCL
10–15	Reserved for future use	VCL
16–18	Slice segment of BLA picture	VCL
19, 20	Slice segment of IDR picture	VCL
21	Slice segment of CRA picture	VCL
22–31	Reserved for future use	VCL
32	Video parameter set (VPS)	non-VCL
33	Sequence parameter set (SPS)	non-VCL
34	Picture parameter set (PPS)	non-VCL
35	Access unit delimiter	non-VCL
36	End of sequence	non-VCL
37	End of bitstream	non-VCL
38	Filler data	non-VCL
39, 40	SEI messages	non-VCL
41–47	Reserved for future use	non-VCL
48–63	Unspecified (available for system use)	non-VCL

U okviru NAL jednice imamo zaglavlje NAL jedinice i korisni sadržaj NAL jedinice. Zaglavlje ima 2 bita i služi

da bi se označila namena korisnog sadržaja jedinice. Na slici 1. prikazano je zaglavlje NAL jedinice.



Slika 1. *Zaglavlje NAL jedinice [2]*

2.2 Otpornost na greške

Za razliku od H.264, glavna tehnika za otpornost na greške u H.265 jeste „sečenje“ („*slicing*“). Ova tehnika zasniva se na VCL NAL jedinicama koje staju u paket podataka i koje skoro da nemaju zavisnost kodovanja od ostalih *slice*-eva u istom frejmu. Na taj način gubitak jednog *slice* ne mora da utiče na kodovanje ostalih *slice*-eva.

3. PROFILI, NIVOI I RANGOVI

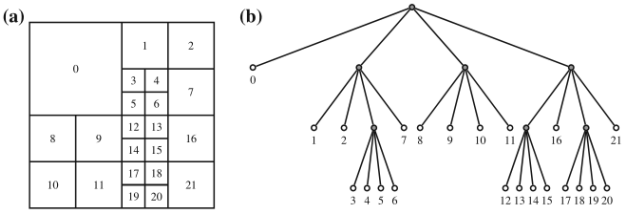
Standard H.264 ima jedan najvažniji nivo „*High*“. Standard H.265 ima tri najvažnija profila: Glavni, Glavni 10 i Glavni mirna slika. Oni podržavaju samo 4:2:0 format odabiranja. Dodatna dubina bita kod profila Glavni 10 omogućava mu da u odnosu na Glavni profil pruži bolji kvalitet videa. H.265 definiše 13 nivoa. Prikazani su u tabeli 2. Maksimalna rezolucija za H.264 jeste 4096x2304. Za H.265 maksimalna rezolucija jeste 7680x4320(8k UHD-2). U H.265 nivoi su podeljeni u dva ranga: Glavni za nivoe 1-3.1 i Visoki za nivoe 4-6.2. (za zahtevnije aplikacije). Novina u H.265 jesu nivoi 6, 6.1, 6.2 koji služe za podržavanje 8k UHD-2 videa. Najveća brzina bita od 800 Mbit/s, postignuta primenom Visokog ranga za prenos jednog 8k UHD-2 *stream*-a, više od tri puta veća je od brzine koju ostvaruje Glavni rang. Glavni rang za prenos jednog 4k-UHD-2 „stream“-a preko Wi-Fi mreže ili interneta visoke brzine, zahteva maksimalnu brzinu od 25 Mbit/s za 30 Hz i 40 Mbit/s za 60 Hz što je dovoljno za pružanje usluge *online* UHD TV. U standardu H.265 izvršena je dopuna ekstenzija opsega sa 19 dodatnih profila. Smanjenje u brzini bita do 7% postiže se za 4:4:4 video, a za RGB video može dostići i 26%. U RGB videu postoji veća korelacija između komponenti što dovodi do većeg smanjenja u brzini bita.

Tabela 2. *Nivoi Glavnog profila [3]*

Level	Max luma picture size (samples)	Max luma sample rate (samples/sec)	Main tier max bit rate (1,000 bits/s)	High tier max bit rate (1,000 bits)	Min comp. ratio
1	36,864	552,960	128	–	2
2	122,880	3,686,400	1,500	–	2
2.1	245,760	7,372,800	3,000	–	2
3	552,960	16,588,800	6,000	–	2
3.1	983,040	33,177,600	10,000	–	2
4	2,228,224	66,846,720	12,000	30,000	4
4.1	2,228,224	133,693,440	20,000	50,000	4
5	8,912,896	267,386,880	25,000	100,000	6
5.1	8,912,896	534,773,760	40,000	160,000	8
5.2	8,912,896	1,069,547,520	60,000	240,000	8
6	33,423,360	1,069,547,520	60,000	240,000	8
6.1	33,423,360	2,005,401,600	120,000	480,000	8
6.2	33,423,360	4,010,803,200	240,000	800,000	6

4. *QUADTREE*

Novina koju H.265 donosi jeste primena particionisanja adaptivnog bloka pomoću „*quadtree*“ strukture koja unapređuje pretragu pokreta. Na slici 2. prikazano je *quadtree* particionisanje. Osnovna jedinica procesiranja u „*quadtree*“ strukturi jeste jedinica kodnog drveta (CTU – *Coding Tree Unit*).

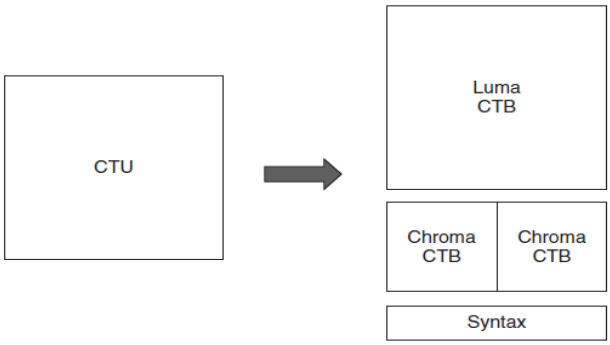


Slika 2. *Ponavljanje “quadtree” particionisanja [4]*

Primena *quadtree* strukture omogućava da frejm može biti particionisan i kodovan primenom blokova različitih veličina (veći blokovi koriste se za iste ili glatke površine; manji blokovi koriste se obično za oblasti sa puno detalja koji imaju visoko kontrastne ivice). Ovo dovodi do efikasnijeg kodovanja i smanjuje vreme kodovanja. H.264 koristi MB veličine 16x16 za particionisanje frejma, a H.265 uvodi blok veličine 64x64 piksela.

4.1. Jedinica kodnog drveta

CTU sadrži tri stavke (vidimo na slici 3.): luminentni blok kodnog drveta, dva odgovarajuća hrominetna bloka i odgovarajuću sintaksu predikcije koja nam daje informaciju o izabranom tipu predikcije.



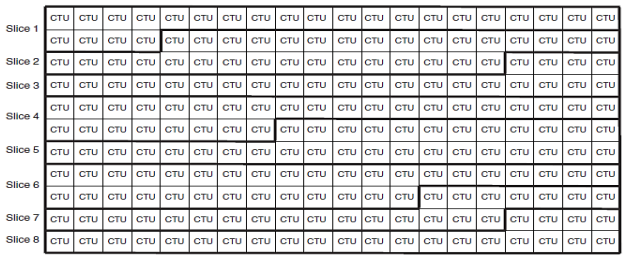
Slika 3. *Komponente CTU-a [5]*

Jedan CTU može se primeniti kao jedan CU ili podeljen u više CU-ova. CU jeste kvadratnog oblika. Može se particionisati u jednu ili više predikcionih jedinica (PU), namenjenih intra tj. inter predikciji i u jednu ili više transformacionih jedinica (TU), namenjenih za transformisanje i kvantizaciju. Jedan CTB (*Coding Tree Block*) blok, određene oblasti frejma, može biti primenjen kao jedan CB ili može biti podeljen na više manjih CB-ova. Veličina CB-a može da varira od veličine CTB-a (maksimalne veličine 64x64) do veličine luminentnih uzoraka 8x8. U standardu H.265, primenom većih CTB-ova (posebno u slučaju kada nema finih detalja ili pokreta u datom CTB-u), poboljšana je kompresija. Korišćenjem manjih CTB-ova dolazi do nagomilavanja *overhead*-ova što negativno utiče na kompresiju. Veličina kvadratnog TB-a u H.265 varira od 4x4 do 32x32, dok H.264 podržava transformacione veličine do 8x8. Za pravougaone TB-ove, H.265 dozvoljava veličine: 32x8, 8x32, 16x4, 4x16.

5. *SLICE*

Posmatrani frejm može biti tretiran kao jedan *slice* ili može biti podeljen na više *slice*-ova. Na slici 4. dat je primer *slice*-a. Preko granice *slice*-eva ne izvode se podaci

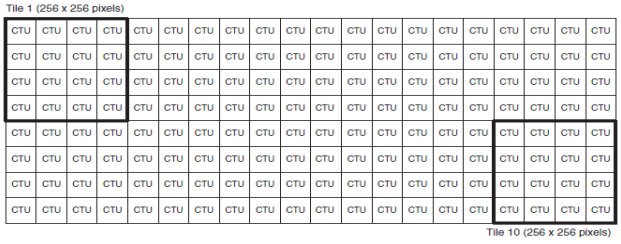
predikcije, podaci ostatka i entropijsko kodovanje, što ih čini samostalnim i omogućava njihov prenos u zasebnim NAL jednicama. Na taj način ostvareno je nezavisno kodovanje *slice*-eva. U slučaju kada je primenjena operacija filtriranja na frejm, može biti potreban prenos informacija preko granice *slice*-a. Takvo „*interslice*“ filtriranje dato je opcionalno. U zavisnosti od tipa *slice*-a, na blokove mogu biti primenjeni kodovi različitih dužina što dovodi do problema sinhronizacije. Standard H.264 rešava ovaj problem primenom jedinstvenog resinhronizacionog markera (0000 0000 0000 0000 1), dok kod standarda H.265 sama struktura podataka u *slice* pomaže u resinhronizaciji i poboljšava otpornost na greške.



Slika 4. *Primer „slice”-eva [5]*

5.1. *Tiles*

Standard H.265 pomoću „*tiles*“ (pločice) deli frejm na rešetku pravougaonih regija. Na slici 5. dat je primer kvadratne pločice. Najmanji broj luminentnih uzoraka koji *tile* mora da sadrži iznosi 256x64. U paralelnom procesiranju *tiles* jesu fleksibilnije od *slices*. Primena *tiles*-ova omogućava nezavisno kodovanje višestrukih pravouganih izvora u datom frejmu (npr. aplikacija slika u slici). Kao kod *slice* i na granicama *tile* dolazi do prekidanja zavisnosti predikcije što omogućava nezavisno procesiranje. Razlika između *tiles* i *slices* jeste ta što je primarna uloga *tiles*-a paralelno procesiranje, a *slice*-a otpornost na greške. *Tiles* imaju važnu primenu i u softverski projektovanoj arhitekturi gde postoje problemi kašnjenja tj. nedovoljne iskorišćenosti procesora, prilikom sinhronizacije procesora ili jezgara nivo CTU-a. Upotreba kompleksnih sinhronizacionih niti prilikom kodiranja i dekodiranja paralelno-procesiranih arhitektura izbegava se primenom *tiles*.



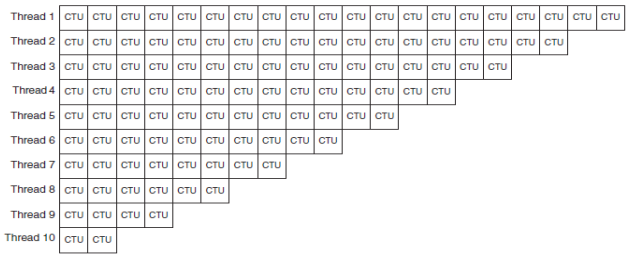
Slika 5. *Primer kvadratnih pločica [5]*

5.2. Paralelno talasno procesiranje WPP

U *slice* i *tile*, zavisnosti u predikcionom postupku mogu preći preko granice blokova što se može negativno odraziti na performanse kodovanja. WPP (*Wavefront Parallel Processing*) omogućava paralelno kodovanje i dekodovanje na nivou *slice*-a bez prekidanja zavisnosti predikcije (kontekst u entropijskom kodovanju koristi se što je češće moguće).

Tehnika „talasa“ (WPP) deli frejm na redove CTU-ova. Svaki red može biti procesiran različitom niti („*thread*“) čim se procesiraju dva CTU-a u redu iznad. Primer Paralelnog talasnog dekodovanja dat je na slici 6.

Tehnika WPP-a postiže bolju kompresiju u odnosu na *tiles* primenom prethodnih podataka u entropijskom kodovanju.



Slika 6. *Paralelno talasno dekodovanje [5]*

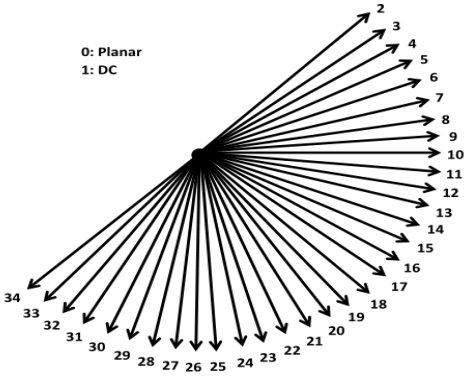
Prednost WPP-a jeste i to što može da izbegne vizuelne artefakte nastale primenom *tiles*.

6. INTRA PREDIKCIJA

H.265 u intra predikciji podržava 35 modova, 33 direkciona, planarni i DC mod. H.264 podržava 8 direkcionih modova, planarni i DC mod. U H.265, na blok veličine 8x8, 16x16 i 32x32 može se primeniti bilo koji mod. Standard H.264 primenjuje 8 direkcionih modova za blokove veličina 4x4 i 8x8, a samo 4 moda za blok veličine 16x16.

6.1. Ugaona predikcija

Koristi se za regije u kojima postoje jake direkcione ivice. Na slici 7. prikazana su 33 predikciona pravca koje podržava H.265. Izbor uglova za predikcione modove izvršen je u skladu sa statistički preovlađujućim uglovima u prirodi. Sa slike 7 vidimo da je fino zrnasto pokrivanje obezbeđeno za skoro vertikalne i skoro horizontalne pravce (oko modova 10 i 26). Za skoro dijagonalne pravce imamo manje gusto pokrivanje.



Slika 7. *Intrapredikcioni modovi [6]*

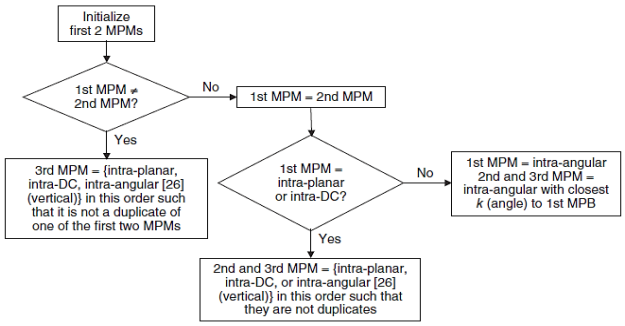
6.2 Planarna predikcija

Planarna predikcija u H.265 može biti primenjena za sve veličine PB-a, dok u H.264 može biti upotrebljena samo za luminentni PB veličine 16x16. Unapređeno je adaptivno filtriranje referentnih uzoraka korišćeno u H.264 za blokove veličine 8x8. Standard H.265 ovo filtriranje može upotrebiti za veličine 8x8, 16x16 i 32x32 u više različitih modova.

6.3 Mod kodovanja

Za luminentne komponente, H.264 u intra predikciji bira jedan mod koji najviše odgovara. Kod H.265 biraju se tri

najpovoljnija moda pri čemu se vodi računa o dupliranju modova (uvode se alternativni modovi). Na slici 8. prikazan je algoritam odabira MPM (*Most Probable Modes*). Za hrominentne komponente, H.265 uvodi intra-izvedeni mod u kome se za hrominentni blok, koristi mod korišćen pri predikciji luminentnog bloka.



Slika 8. *Odabir MPM [6]*

7. INTER PREDIKCIJA

U standardu H.265 podržano je više particija u inter predikciji, nego u intra predikciji. Particije mogu biti kvadratne ili pravougaone, pri čemu pravougano particionisanje pruža precizniju predikciju pokreta. Asimetrično particionisanje pokreta (AMP – *Asymmetric Motion Partitioning*), primenjeno na granice nepravilnog bloka, poboljšava efikasnost kodovanja. H.265, primenom dužih tap filtara (7/8 tap filtri) pri interpolaciji frakcionog uzorka, postiže veću preciznost nego standard H.264. Predikcija vektora pokreta (MVP – *Motion-Vector Prediction*) značajno je poboljšana kod H.265. Mod spajanja blokova olakšao je signalizaciju podataka pokreta primenom već dekodiranih blokova. Standard H.265, u cilju efikasnije obrade fleksibilne strukture bloka nastale primenom „*quadtree*“ particionisanja, uvodi naprednu predikciju vektora pokreta (AMVP – *Advanced MVP*).

8. TRANSFORMACIJA, SKALIRANJE KVANTIZACIJA

Celobrojne transformacije bloka primenjene u H.265 daju preciznije aproksimacije DCT-a (*Discrete Cosine Transform*) u odnosu na transformacije iz H.264. Osnovni vektori transformacija u H.265 imaju jednaku energiju, tako da nije potrebna kompenzacija različitih normi iz H.264. Celobrojna aproksimacija DST (*Discrete Sine Transform*) alternativno može biti upotrebljena prilikom obrade intraprediktovanih 4x4 luminentnih ostataka TB-ova. Pogodnija je za oslabljenu statističku korelaciju, jer je tada udaljenost od graničnih uzoraka veća. U H.265 nije potrebna operacija preskaliranja (dekvantizacija) iz H.264 što dovodi do smanjenja srednje veličine memorije. Za transformacione veličine 32x32 ostvarene su značajne uštede memorije. U standardu H.265 primenjena je uniformna rekonstrukciona kvantizacija (URQ – *Uniform Reconstruction Quantization*).

9. ZAKLJUČAK

H.265 (HEVC) predstavlja očekivano poboljšanje u odnosu na standard H.264. Unapređuje tehnike korišćene u H.264 i uvodi nove tehnike. U H.265 novi set, VPS, predstavlja unapređenje u odnosu na H.264 višeslojne ili „*multiview*“ ekstenzije. Za podržavanje 8k UHD-2 videa H.265 uvodi nivoe 6, 6.1, 6.2. Primenom „*quadtree*” particionisanja postiže se efikasnije kodovanje i smanjuje vreme kodovanja u odnosu na tehniku particionisanja iz H.264. Prilikom kodovanja i dekodovanja UHD/HD videa svi alati paralelnog procesiranja visokog nivoa standarda H:265 postaju efikasniji. U H.265, paralelno procesiranje „talasima“ i „*tiles*“-ima, dato je opciono što omogućava da dekoderi koji imaju ograničene procesorske resurse mogu da dekoduju video bez paralelnog procesiranja. Mogućnost primene 33 predikciona pravca za intra predikciju, u standardu H.265, povećava efikasnost kodovanja u odnosu na neke druge metode kompresije bez gubitka (npr. JPEG). Vrednosti poboljšanja su 20%-50%. Za hrominentne komponente, H.265 uvodi intra-izvedeni mod u kome se za hrominentni blok, koristi mod korišćen pri predikciji luminentnog bloka. Standard H.265 poboljšava predikciju vektora pokreta uvođenjem AMVP. Pri interpolaciji frakcionog uzorka, H.265 ostvaruje veću preciznost nego H.264.

10. LITERATURA

[1] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wigand, „Overview of the High Efficiency Video Coding (HEVC) Standard“, IEEE Transactions on Circuits and Systems for Video Technology, 22(12), 1649-1668, (Dec. 2012)

[2] Thomas Schierl, Ye-Kui Wang, Miska M. Hannuksela, Stephan Wenger, „System layer integration of High Efficiency Video Coding“, IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, (Dec. 2012)

[3] K. R. Rao, Do Nyeon Kim, Jae Jeong Hwang, „Video coding Standards AVS China, H.264/MPEG-4 PART 10, HEVC, VP6, Dirac and VC-1“, ISBN 978-94-007-6742-3, 2014.

[4] Mathias Wien, „High Efficient Video Coding: Coding Tools and Specification“, ISBN 978-3-662-44276-0, 2015.

[5] Benny Bing „Next-generation video coding and streaming“, Online ISBN 9781119133346, John Wiley&Sons 2015.

[6] Guilherme Correa, Pedro Assuncao, Luciano Agostini, Luis A. da Silva Cruz, „Complexity-Aware High Efficeiency Video Coding“, ISBN 978-3-319-25778-5, 2016.

Kratka biografija:

Mara Janković, rođena je u Senti 1981. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Telekomunikacije odbranila je 2019.god.
kontakt: jankovicmara@gmail.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.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.39
DOI: https://doi.org/10.24867/07BE00Popov

IPTV SISTEMI NA REGIONALNOM TRŽIŠTU

IPTV SYSTEMS ON THE REGIONAL MARKET

Tomislav Popov, Željen Trpovski, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je opisano šta je IPTV i istorijat njenog nastanka, zatim su predstavljene glavne tehničke karakteristike i osnovni parametri vezani za IPTV. Jedan deo rada je posvećen pojavi i razvoju IPTV u Srbiji i regionu.*

Ključne reči: *IPTV, Digitalna televizija, IPTV Srbija*

Abstract – *This paper analyses IPTV and its history. It decsribes some of the basic characteristics and tehnical parameters of IPTV. Part of the paper is devoted to the development of IPTV in Serbia and region.*

Keywords: *IPTV, Digital television, IPTV Serbia*

1. UVOD

Svedoci smo stalnog tehnološkog napretka, u svim aspektima života čovek traži neko novo bolje rešenje. Televizija kao servis koji služi da informiše i zabavi čoveka prati taj trend takođe. Digitalizacijom televizija je napravila veliki korak u poboljšavanju kvaliteta. Razvojem interneta omogućena je dostupnost informacijama bilo gde na zemaljskoj kugli. Digitalni tv signal dobija novu mogućnost za distribuciju posredstvom internet protokola (IP) i takva televizija naziva se Internet Protokol Televizija (IPTV).

Pored toga što je televizija sada povećala svoju dostupnost, ovakav vid distribucije donosi novu, interaktivnu televiziju. Ovo znači da za razliku od tradicionalnih vidova distribucije (kada informacije odnosno signal idu samo u jednom smeru od distributera ka korisniku), IPTV ima dvosmeran tok informacija odnosno postoji povratni signal od korisnika ka distributeru.

Upravo ta neposredna komunikacija između korisnika i distributera otvara neke nove mogućnosti čini veliku razliku između televizije pre i posle nastanka IPTV.

2. POJAM IPTV, PREDNOSTI I NEDOSTACI

Internet Protokol Televizija (IPTV) je sistem pomoću koga se digitalna televizija dostavlja korisnicima preko Internet Protokol (IP) mreže. Digitalizacijom televizija je dobila mogućnost da se za njenu distribuciju može koristiti internet mreža. Ovakav vid distribucije televiziji pruža mnogo novih mogućnosti i dodatnih servisa koje dosadašnji vidovi dostribucije nisu mogli da pruže.

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

2.1. Prednosti IPTV

Od mnogo razloga koji su doveli do toga da se izabere IP mreža za prenos video sadržaja tri su najpopularnija: fleksibilnost IP mreže, mala cena koštanja i velika pokrivenost koju IP mreža pruža širom sveta.

2.2. Nedostaci IPTV

Prva stvar koja nas dovodi u dilemu jeste ekonomske prirode, pokazalo se kroz istoriju korišćenja Interneta da su mnoge stvari besplatne i kao takve dostupne svima.

Drugi razlog je čisto tehničke prirode i tiče se težine zahteva da video signal bude kontinualan i u vremenu konstantan i prilagođen prenosu preko Interneta.

Treći razlog tiče se kombinovanja prenosa video signala koji ima velike zahteve u pogledu resursa i ostalih stvari koje se transportuju Internetom i kako odrediti koja od njih će imati prioritet.

3. KARAKTERISTIKE IPTV

Od svih karakteristika koje opisuju IPTV ovde bih istakao one koje prave razliku između IPTV i drugih vidova televizije. Tako je to usluga VOD, odnosno video na zahtev ova usluga je moguća zbog dvosmernog protoka informacija korisnik servisni centar. VOD je usluga koju pruža IPTV a sastoji se od mogućnosti naručivanja video sadržaja (serija, filmova, dokumentarnog programa), koji se naplaćuje posebno po tačno određenoj tarifi i koji stoji na raspolaganju korisniku određeni vremenski period u kome on može da ga pregleda neograničen broj puta.

VOD funkcioniše putem kataloga i naručivanja video sadržaja tako da je neophodno da korisnik može da vidi listu materijala koji je moguće naručiti. Još jednu mogućnost treba ovde napomenuti a to je Personal video recorder (PVR) odnosno lični video rekorder koji omogućuje snimanje sadržaja koji korisnik može unapred zakazati, materijal se može sačuvati bilo na STB uređaju, USB memoriji ili na nekom hard disku u sistemu. Snimljeni sadržaj se naknadno može pregledati sa svim opcijama poput premotavanja pauziranja itd

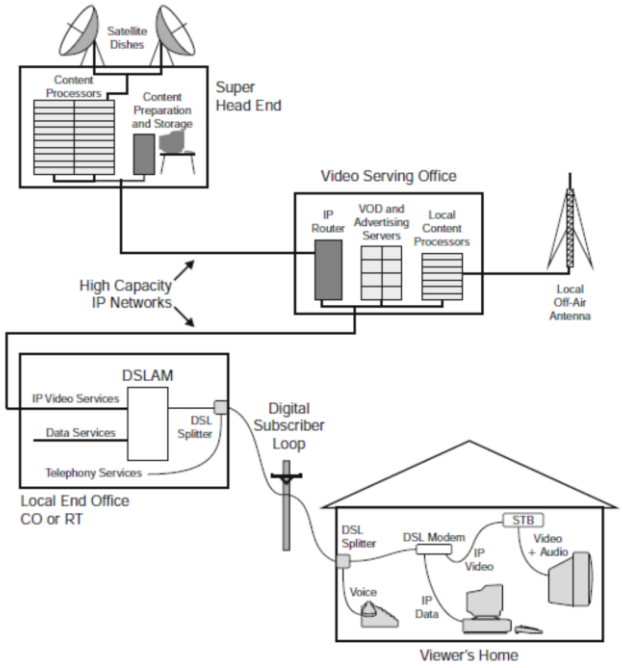
Triple play je usluga koja obuhvata pored televizije, internet vezu brzog protoka i telefonsku liniju. IPTV kao tehnologija dostavljanja televizijskog sadržaja putem IP mreže, pošto i sama koristi internet jeste u mogućnosti da korisnicima ponudi i internet konekciju za prenos podataka a budući da u delu svog sistema koristi i već postojeću infrastrukturu fiksne telefonije nameće se kao vrlo zgodno da se sve te usluge obuhvate jednim paketom koji se dostavlja istim signalom. Fiksna telefonija ovde dobija dodatne mogućnosti kao što su preusmeravanje poziva, identifikacija poziva itd.

4. TOPOLOGIJA IPTV MREŽE

Na slici 1 prikazana je tipična struktura IPTV mreže. IPTV koristi već postojeću infrastrukturu, internet mrežu i mrežu koju telefonske kompanije imaju postavljenu za potrebe fiksne telefonije da bi svojim korosnicima dostavila na stotine tv kanala

Super Head End (SHE) treba da omogući uslugu velikom broju korisnika tako što obrađuje video sadržaj koji zadovoljava potrebe svih korisnika, to je mesto gde se iz različitih izvora sakuplja tv sadržaj koji se zatim konvertuje u oblik pogodan za transport preko IP mreže. SHE je početak IPTV mreže.

Taj sadržaj se zatim prosleđuje do Video serving office (VSO), koji je zadužen da dostavi sadržaj za teritoriju određenog grada ili geografskog područja. VSO pored sadržaja koji dobija od SHE dodaje lokalni sadržaj bilo lokalne tv stanice ili neku vrstu marketinga. Dodatni video sadržaj takođe mora biti konvertovan u pogodan format.



Slika 1. Tipična struktura IPTV mreže *[1]*

Na ovom mestu se već formira IPTV sadržaj pogodan ne samo za dalji transport do Central office (CO) nego i do samih korisnika. Na ovom mestu formira se signal za svakog korisnika posebno.

Ako isti sadržaj zahteva veći broj korisnika onda se ovde pravi kopija i šalje se više istih signala.

CO su sledeće u nizu stanica koje su zadužene za manje oblasti, ovde se nalazi oprema koja omogućuje transport IPTV sadržaja preko DSL (digital subscriber line)do krajnjih korisnika.

Na kraju dolazi DSL modem koji se nalazi kod svakog korisnika i treba da omogući razdvajanje fiksne telefonije internet konekcije i IPTV sadržaja koji se šalje na STB uređaj. STB uređaj podržava mnoge mogućnosti koje IPTV pruža, dekodira video sadržaj I omogućuje njegovo gledanje na tv prijemniku, pruža grafički prikaz za prebacivanje kanala i korišćenje opcija koje IPTV nudi, i on takođe mora biti takav da je kompatibilan sa svim vrstama tv prijemnika inače može da se dogodi da IPTVusluga ne funkcioniše kako treba.

5. KOMPRIMOVANJE AUDIO I VIDEO SADRŽAJA

Video i audio signal koji se koristi kod IPTV je uvek komprimovan. Komprimovanje znači redukovanje broja bitova koji su potrebni da bi reprezentovali neku sliku. Dobro urađeno komprimovanje znači veću fleksibilnost, više izbora sa postojećim kapacitetima koji mogu poneti više podataka ili postići veći kvalitet podataka ili oba. Ako pogledamo televizijski program, tamo gde je nekad bio prenošen jedan analogni signal sada pomoću komprimovanja može biti prenešeno i stotine kanala što znači mnogo ekonomičnije prenošenje televizijskog programa do korisnika.

Komprimovan signal može biti prenošen i kod mreža sa manjom brzinom prenosa, ovo kod onih korisnika koji imaju spore internet konekcije nekada može značiti i mogućnost uopšte preneti video sadržaj ili ne. Drugi razlog je taj što u postojeći kapacitet protoka može stati više komprimovanih signala, što je važno za IPTV zbog ograničenja u prenosu na određenu daljinu. Na primer za ADSL2+ ima ograničenje od 10 Mbps na daljinu od 2750m, uz normalne tehnike komprimovanja 10Mbps je dovoljno za dva do cetiri SD signala ili jedan HD i nekoliko SD signala.

Kako tehnike komprimovanja napreduju moguće je više materijala progurati uz isti protok. Grubo rečeno, jedan nekomprimovani HD signal zauzima 1.5 Gbps što je otprilike 1000 puta više od standardne ADSL veze, a to znači da je bez komprimovanja jedan HD signal praktično nemoguće preneti u okviru jedne standardne IPTV mreže.

Naravno iako ima mnoge prednosti komprimovanje nužno unosi i neke nepovoljne stvari u proces obrade i prenosa video i audio materijala. Komprimovanje unosi kašnjenje u prenos video i audio signala i to i kod komprimovanja i kod dekomprimovanja. To se događa zbog toga što je potrebno sačuvati nekoliko frejmova koje treba uporediti sa referentnim signalom. Ponekad komprimovanje može biti izuzetno teško zbog smetnji ili nepravilnosti u signalu pa postoje poteškoće prilikom procesa komprimovanja.

IPTV signal je moguće komprimovati na više načina a jedan od zastupljenijih je MPEG sistem komprimovanja odnosno MPEG standardi za kompresiju.

6. KVALITET I SIGURNOST IPTV SIGNALA

Kvalitet i sigurnost su važne osobine svakog prenosnog sistema pa tako i sistema za prenos video sadržaja kao što je IPTV. Da bi korisnici bili zadovoljni a samim tim spremni da plaćaju uslugu i da koriste i druge usluge u okviru servisa što znači još finansijskih sredstava, da bi adekvatno bio prenet signal televizijskih kuća i da bi funkcionisao marketing neophodan uslov je da se tokom prenosa održi kvalitet signala. Sigurnost je važna da bi korisnici mogli da gledaju samo onaj sadržaj na koji su pretplaćeni odnosno za koji su autorizovani i da bi se sprečilo kopiranje ili umnožavanje sadržaja koji je predviđen samo za gledanje.

Što se tiče kvaliteta signala ona se ogleda u dobroj sinhronizaciji audio i video signala, minimizaciji grešaka u prenosu koje se ogledaju u što manjem broju prekida kako video tako i audio signala,zato što svaki prekid ili izgubljeni paket signala se manifestuje ili kratkotrajnim zamrzavanjem slike ili slikom koja izgleda kao da je u mnogo manjoj rezoluciji, a što se tiče zvuka pokazalo se da je uho mnogo osetljivije odnosno da se kratkotrajni prekid audio signala mnogo lakše uoči nego kratkotrajni prekid video signala.

U kvalitet same usluge spada i kašnjenje prilikom prebacivanja programa ili aktiviranja neke usluge pošto je jasno da svako čekanje iritira korisnika, brzina odziva zavisi od više faktora kao što su kvalitet dolaznog signala od strane TV kompanija, od kvaliteta komprimovanja, od kapaciteta prenosne mreže, od kvaliteta kodera i dekodera i dr.

Što se tiče sigurnosti, koriste se principi zaključavanja sadržaja i korišćenja lozinke za otključavanje istog.

7. ISTORIJAT IPTV

ABC World News Now je prvi televizijski šou koji je emitovan preko interneta, 1994. godine. Za to je korišćen CU-SeeMe video konferencijski softer.

Termin IPTV prvi put se pojavio 1995. godine sa stvaranjem programa napisanog od strane Džudit Estrina i Bil Kariko. Tim softverom su uspostavljena pravila i formiran internetski proizvod pod nazivom IP TV. On je MBONE kompatibilna Windows i Unix bazirana aplikacija koja je podržavala jedno i multi izvorni audio/video saobraćaj, počevši od niskog do DVD kvaliteta, koristeći unikast i IP multikast transportne protokole u realnom vremenu.

Softver su prvenstveno napisali Stiv Kasner, Karl Auerbah i Cha Chi Kuan. Proizvod je kupljen od strane Cisko Sistemi 1998. godine, koja zadržava IP TV zaštitni znak.

Internet radio kompanija AudioNet je počela prvi live webcast sa VFAA-TV i KCTU/LP u januaru 1998. godine.

Kingston Comunication, regionalni telekomunikacioni operater u Velikoj Britaniji pokrenule su KOMPLET (Kingston interaktivna televizija), IPTV preko ADSL širokopojasne veze u septembru 1999. godine, nakon sprovođenja raznih TV i VOD proba. 2001. godine operater je dodao i VOD servis. Kingston je bila jedna od prvih kompanij koja uvodi IPTV i VOD preko ADSL.

Godine 1999. NBTel (sada poznatija kao Bel Aliant) je prvi komercijalno primenio Internet protokol televiziju preko digitalne pretplatničke linije (DSL) u Kanadi. Pomoću Nokia 7350 DSLAM i mrežne infrastrukture je kreirao iMagic televiziju. Usluga se pojavila na tržištu pod brendom VibeVision u Nju Bransviku, a kasnije se proširio na Novu Škotsku početkom 2000., nakon formiranja Aliant. iMagic televizija je kasnije prodata Alkatelu...

8. IPTV U SRBIJI I REGIONU

IPTV servis je u Srbiji pokrenut 15. oktobra 2008. godine, od strane kompanije Telekom Srbija pod nazivom OPEN IPTV. Servis je prva tri meseca radio u probnom test režimu i bio je dostupan samo određenom ograničenom broju korisnika koji su unapred izabrani. Od 1. januara 2009. godine OPEN IPTV ulazi u komercijalnu upotrebu. U 2010. godini broj korisnika je bio oko dvadeset tri hiljade a prihod u 2009. godini je bio rekordan od oko milijardu evra.U tabeli 1 prikazan je broj korisnika Telekoma Srbije po godinama.

Od strane Telekoma Crne Gore koji se nalazi u vlasništvu Mađer Telekoma ponuđen je IPTV servis pod nazivom Extra TV. Broj korisnika ovog servisa krajem 2008. godine prešao je osamnaest hiljada i ostvaren je prihod od oko 123 miliona evra u 2009. godini.

BH Telekom je 14. januara 2009. godine potpisao ugovor sa Smart Com Slovenija o uvođenju integrisanog sistema IPTV i IP telefonije u BiH. Broj korisnika IPTV servisa je do polovine 2010. godine bio petnaest hiljada a ostvareni prihod u 2009. godini oko 303 miliona evra.

U Hrvatskoj je T-com lansirao IPTV servis Max TV sa 65 kanala plus deset novih kanala, usluge video na zahtev i digitalnog video rekordera. U Hrvatskoj je 2010. bilo 250 hiljada korisnika a u 2009. godini ostvarili su prihod od oko 546 miliona evra.

Deutsche Telekom kao vlasnik Makedonskog Telekoma je od decembra 2008. godine otpočeo sa realizacijom IPTV servisa u Makedoniji pod nazivom Max TV.

IPTV je u Republiku Srpsku stigao početkom 2010. godine nakon kupovine većinskog paketa Telekoma Srpske od strane Telekoma Srbija.

Najveći operater u Sloveniji Telekom Slovenija pokrenuo je IPTV servis pod nazivom SiOL TV još 2003. godine. U ponudi su 130 kanala, 15 radio stanica, videoteka sa preko 700 naslova kao i zabava u vidu logičkih igara, igara sa kartama, poteznih i arkadnih igara. SiOL TV nudi prijem IPTV servisa na više prijemnika a 2006. je eksperimentalno prenosio svetsko prvenstvo u visokoj rezoluciji. Broj pretplatnika u 2009. godini je bio oko 110 hiljada. Drugi operator koji ne koristi klasičnu telefonsku infrastrukturu i ADSL tehologiju nego optičku mrežu sprovedenu do brojnih korisnika je T-2. Zbog optičke mreže oni bez problema mogu da omoguće praćenje HD kanala na većem broju prijemnika, u ponudi imaju oko 160 kanala od kojih šest visoke rezolucije. Postoje još dva operatera sa ponudama od 90 kanala Amis i od 100 kanala Tuš.

Tabela 1. Tabelarni prikaz broja korisnika Telekoma Srbije[2,3]

Godina	*2011*	*2012*	*2013*	*2014*	*2017*	*2018*
Broj u hiljadama	*118*	*175*	*244*	*314*	*1730*	*1840*

9. ZAKLJUČAK

Jedna od osnovnih odlika modernog društva jeste stalni tehnološki napredak. Televizija kao sastavni deo svakodnevnog života, zabava i razonoda od samog njenog nastanka doživljava takođe svojevrsni tehnološki preobražaj. Ona antenska klasična televizija praktično isčezava. Prva značajna transformacija bila je digitalizacija televizijskog signala, što je otvorilo vrata za brzi napredak kako kvaliteta tako i ponude televizijskog sadržaja.

Digitalizacija video i audio signala otvorila je uz kombinaciju sa internet protokolom za prenos informacija mogućnost za stvaranje IPTV. Pored IPTV i kablovska televizija ima u ponudi na stotine kanala ali je tehnološki IPTV u prednosti. Prenošenjem samo jednog kanala IPTV ima mogućnost da više unapredi kako kvalitet svog signala tako i brzinu odziva na korisnikove potrebe.

To nas dovodi do druge značajne karakteristike a to je stalna i neposredna interakcija korisnika i servisa putem zadavanja zahteva za promenu kanala ili za pokretanje nekog od mnogih servisa koje IPTV nudi. Pošto svoj servis pruža preko mreže fiksne telefonije koja je dobro razvijena, u mogućnosti je da objedini ponudu fiksni telefon, internet i televizija a i da iskoristi već postojeću bazu korisnika.

Svedoci smo da je na polju zabave savremeno društvo možda i najviše napredovalo. Primenom takozvanih pametnih telefona zabavni sadržaj u svakom obliku kao i internet konekcija dostupni su na svakom koraku.

Nekada je televizija bila jedina stvar koja je služila za zabavu i kao medij preko kojeg su se širile opšte informacije u obliku informativnog programa. Sada je tu internet kao izvor informacija iz svih sfera života, a što se tiče zabave tu su mnoge društvene mreže, specijalizovani kanali za pregledanje muzičkih video spotova, sajtovi za razmenu video i audio sadržaja itd.

IPTV po kvalitetu i kvantitetu televizijskog programa i svih pratećih servisa, kao i po dostupnosti može biti možda i najbolji televizijski servis do sada, gledano kroz istoriju. Ali da li je moderno društvo kome su sada dostupne informacije i zabava na svakom koraku zahvaljujući pametnim telefonima koji su sastavni deo života gotovo svakog od nas i dalje zainteresovano za koncept televizije ili je on prevaziđen, pokazaće vreme.

10. LITERATURA

[1] IPTV and Internet Video
Expanding the reach of television broadcasting, Wes Simpson and Howard Greenfield.
[2] www.mts.rs (konsolidovani godišnji izveštaj o poslovanju za 2017 i 2018 godinu.)
[3] www.statista.com

Kratka biografija:

Tomislav Popov rođen je u Kuli 1979. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva odbranio je 2019.god.
kontakt: popovtomislav@gmail.com

Željen Trpovski rođen je u Rijeci 1957. godine. Doktorirao je na Fakultetu tehničkih nauka 1998. god. Oblast interesovanja su telekomunikacije i obrada signala.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.39:681.7.068
DOI: https://doi.org/10.24867/07BE00Avramovic

PRIMOPREDAJNICI ZA PASIVNE OPTIČKE MREŽE

TRANSCEIVERS FOR PASSIVE OPTICAL NETWORKS

Marija Avramović, Željen Trpovski, Dejan Nemec, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Dat je prikaz pasivnih optičkih mreža i njihove međusobne razlike. U ovom radu opisane su tehnologije primopredajnika, primopredajnički monitoring i princip rada OTDR-a.*

Ključne reči: *PON sistemi, EPON, GPON, OTDR*

Abstract – *Passive optical networks and their differences are presented. This thesis describes transceiver technologies, transceiver monitoring and the operating principle of OTDR.*

Keywords: *PON systems, EPON, GPON, OTDR*

1. UVOD

Optički primopredajnici veoma su značajni za optičke komunikacije, jer obavljaju fundamentalan zadatak pretvaranja optičkog signala u električni i obrnuto. Koriste poluprovodničke lasere i detektore koji rade u sprezi sa integralnim kolima. Za FTTH (*Fiber To The Home*) sisteme primopredajnici predstavljaju ključnu komponentu i kao takvi trebaju se razvijati sa što većim performansama, da prevazilaze oganičenja u stvarnom svetu i budu što jeftiniji.

Mreže na bazi bakra, poput kablovskih modema ili digitalnih pretplatničkih mreža DSL (*Digital Subscriber Line*), ne pružaju toliki propusni opseg i potrebnu razdaljinu za prenošenje multimedijalnih servisa poput govora, podataka i video programa (IPTV; HDTV). Zato su pasivne optičke mreže isplativo i sigurno rešenje ovog problema.

PON (*Passive Optical Network*) sistem sastoji se iz optičkog linijskog terminala (OLT – *Optical Line Termination*), višestrukih mrežnih jedinica (ONU – *Optical Network Unit*) i pasivnog optičkog splitera (razdelnika), koji spaja OLT sisteme sa ONU sistemima.

Ovaj rad baviće se tehnologijom optičkih primopredajnika i njihovom primenom u modernom okruženju koje postavlja sve veće standarde po pitanju brzine protoka sve većih količina podataka.

Dat je prikaz pasivnih optičkih mreža i njihove međusobne razlike. Opisane su i tehnologije primopredajnika, primopredajnički monitoring i princip rada OTDR-a (*Optical Time Domain Reflectometer*).

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

2. ZAHTEVI PON SISTEMA

Postoje tri tipa PON-a, BPON (*Broadband PON*), GPON (*Gigabit PON*) i EPON (*Ethernet PON*). EPON podržava najmanje 16 splitova vlakana.
U tabeli 1. ilustrovan je odnos budžeta snage i sposobnost deljenja tipova PON-a. Maksimalni odnos splita PON-ova nije definisan, jer zavisi od gubitaka vlakana i sposobnosti fizičkog sloja. Budžeti optičkih linkova određuju se individualnim aktivnim komponentama prodavaca.

Tabela 1. O*dnos budžeta snage i sposobnost deljenja (splita) tri tipa PON* [1]

Type & protocal	Split ratio	Power budget
BPON	32 max	Channel loss (only[b]) Class A optics: 20 dB Class B optics: 25 dB Class C optics: 30 dB
GPON	64 max	Channel loss (only[b]) Class A optics: 20 dB Class B optics: 25 dB Class C optics: 30 dB
EPON	16 nominal 32 permitted	PX10 US: 23 dB PX10 DS: 21 dB PX20 US: 26 dB PX20 DS: 26 dB

2.1. Specifikacije fizičkog sloja

Specifikacije fizičkog sloja zavisne su od fizičkog medijuma (PMD – *Physical Medium Dependent*) koji uključuje opto/elektronske konvertore i CDR-a (*Clock and Data Recovery*). Fizički sloj šalje podatke do fizičkog medijuma i obratno. Sloj fizičkog medijuma označava optičke primopredajnike i prijemnike sa snagom prenosa i vrednostima osetljivosti prijemnika za svaki budžet snage i brzinu prenosa.

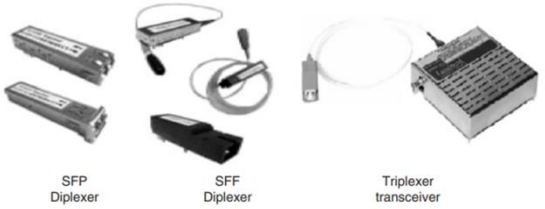
2.2. Vremenski zahtevi *burst* moda (u naletima)

Višestruki *upstream* pristup i ponašanje *burst* moda zahtevaju od ONU transmitera da generišu signale vremenski ograničene unutar alociranih vremenskih slojeva dodeljenih od strane MAC (Medium Access Control) sloja. Ovo znači da snaga lasera mora ostati stabilizovana unutar vremenskog okvira u kojem se signal šalje dok se vremenski slot ne završi. ONU prijemnik mora biti neaktivan i ne sme slati ni najmanji šum tokom alociranja signala na drugi ONU prijemnik, u suprotnom, doći će do unakrsnih razgovora i ometanja *upstream* servisa. Ovakav pristup zahteva brzo menjanje ONU predajnika.

3. TEHNOLOGIJE PRIMOPREDAJNIKA

Optički primopredajnici klasifikuju se prema talasnim dužinama, dometu, brzinama podataka, vrsti pakovanja, temperaturnim opsezima, električnim i optičkim interfejsom i drugo. Postoje dva standardna tipa primopre-

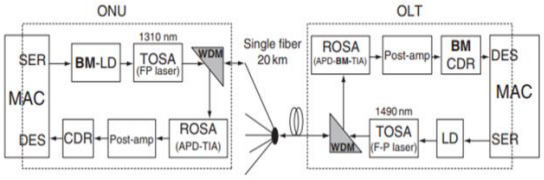
dajnika: dupleksni i tripleksni primopredajnici. Neke od primera prikazani su na slici 1. Kod dupleksnih primopredajnika talasne dužine su određene industrijskim standardima (1.310 nm za *upstream* smer, 1.490 nm za *downstream* smer). Kod tripleksnih se talasna dužina od 1.550 nm alocira za analogno emitovanje videa u *downstream* pravcu.



Slika 1. *Razni PON dupleksni i tripleksni primopredajnici za optičke pristupne sisteme* [2]

3.1. Izrada blokova primopredajnika

ONU strana sastoji se od *downstream* i *upstream* predajnika, dok se OLT strana sastoji od *downstream* i *upstream* prijemnika. *Upstream* ONU sastoji se od laserskog pogona *burst* moda i FP (*Fabry-Perot*) lasera u formi prenosa optičkog podsklopa (TOSA – *Transmitter Optical Sub Assembly*) što je prikazano na slici 2. OLT prijemnik sastoji se od PIN ili lavinske fotodiode u prijemnom optičkom podsklopu (ROSA - *Receiver Optical Sub Assembly*), ograničavajućeg postpojačavača (*post-amp*) i jedinice CDR u *burst* modu.



Slika 2. *Struktura PON primopredajnika [3,4]*

3.2. Optički uređaji prenosa i prijema

Optički primopredajnički uređaji sastoje se od DFB (*Distributed Feedback Laser*) lasera i F- laserske diode. DFB laseri veoma su pogodni za OLT uređaje, jer podnose visoke temperaturne opsege i ne zahtevaju hlađenje. Imaju i veliku brzinu odziva od 0,12 ns, što je vreme podizanja i padanja (20-80%) laserskog zraka. FP laseri koriste se najviše sa ONU strane primopredajnika da bi se smanjila cena sistema.

3.3. Bidirekcionalno optičko podsklapanje (BOSA)

Tehnologija koja se koristi za razvijanje bidirekcionalnog optičkog podsklopa (BOSA – *Bi-Directional Optical Sub Assembly*) bazirana je na *bulk-optic* tehnologiji sklapanja. Ova tehnologija suočava se sa ozbiljnom konkurencijom od strane novih naprednih tehnologija koje su zasnovane na visoko integrisanim planarnim svetlosno-talasnim kolima (PLC) i automatizovanom pasivnom sklopu.

3.4. *Bulk-optic* tehnologija

Konvencionalni dvotipni-TO-CAN BOSA sačinjen je od LD CAN, PD CAN i WDM filtera. U ovoj konfiguraciji, LD i PD aktivno se poravnjavaju sa SMF-om, nezavisno.

3.5. Planarna svetlosno-talasna kola (PLC)

Planarna svetlosno-talasna kola prave se istom tehnologijom procesiranja kojom se prave IC-ovi. Kod PLC pristupa, optički signali prolaze kroz talasovode čipa. Postoje dva konkurentna tipa PLC pristupa za FTTH tržište: eksterni filter PLC-ova i ugrađeni filter PLC-ova.

3.6. Moduli PON primopredajnika

Iz EPON/GPON kombatibilnosti sistema i pogleda na ponovno korišćenje, primopredajnički moduli mogu se podeliti na blokove BOSA, električni podsklop (ESA), kućište i kontrolu temperature. BOSA moduli moraju zadovoljiti optičku snagu predajnika, osetljivost prijemnika, itd. Razmatranja sa ESA strane uključuju *burst* mod upravljačke performanse zavisnog fizičkog medija (PMD), prihvatljivu sistemsku kontrolu signala, veliku osetljivost prijemnika i dinamički opseg snage prijemnika. ONU primopredajnički modul sastoji se od kućišta, LC ili SC konektora i štampanog kola, OSA i upravljačkog kola. OSA je na bazi mikro optike i sadrži optički *band-pass* filter, foto-detektor (PD), električno trans-impedansno pojačavačko (TIA) integralno kolo na strani prijemnika, FP lasersku diodu i monitor PD na strani predajnika. WDM (*Wavelength-Division Multiplexing*) multipleksira predajnik i prijemnik na jedno vlakno.

4. ELEKTRONIKA SA *BURST* MODOM

PON sistemi zahtevaju specijalni prenos i prijem *burst* moda kao jedan od ključnih tehnologija za *upstream* pravac. Ključne komponente takvog PON sistema su predajnik sa *burst* modom unutar ONU-a, koji se nalazi na pretplatničkom kraju i prijemnici, koji se realizuju upotrebom IC-ova.

4.1. Konvencionalni podaci naspram podataka *burst* moda

Prilikom slanja podataka uz pomoć *burst* moda odnos logičkih nula i jedinica i intervala između logičkih promena nije ograničen. Kod konvencionalnog, kontinualnog slanja odnos logičkih jedinica i nula je balansiran i interval između bilo koja dva logička simbola je strogo ograničen. Konvencionalni predajnici i prijemnici jedino su pogodni za kontinualni prenos podataka, dok se koristi naizmenično spajanje.

4.2. Predajnik koji koristi *burst* mod (BM Tx)

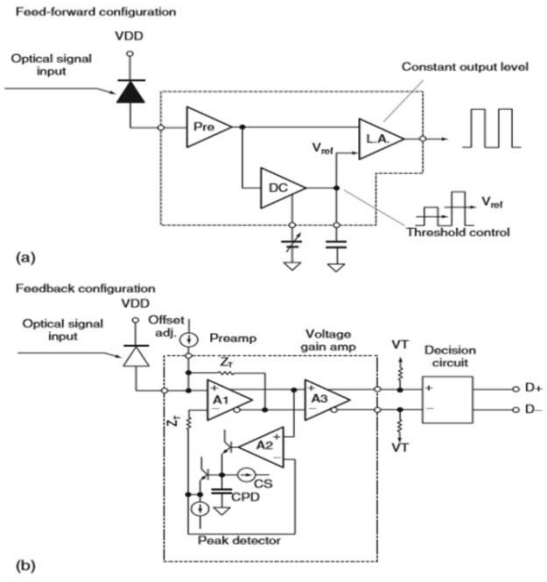
Predajnik BM-Tx kod ONU-a koristi lasersku diodu koja se nalazi u TOSA ili BOSA paketu i lasersku upravljačku diodu integralnog kola (LDD). Upravljačko integralno kolo mora da ima malu potrošnju snage pošto se mora pokrenuti uz pomoć rezervne baterije tokom nestanaka struje.

4.3. Prijemnik koji koristi *burst* mod (BM Rx)

Prijemnici koji koriste *burst* mod mogu se podeliti na dve kategorije prema njihovim strukturama: *"feedforward*" konfiguracija i "*feedback*" konfiguracija.

U *feedforward* implementaciji, može se koristiti konvencionalno jednosmerno spojeni predpojačavač. Primljeni signal prvo se pojačava, zatim deli na dve grane. Prva grana izlaza iz predpojačavača jednosmerno se spaja na diferencijalni pojačavač. Druga grana se *feedforward*-uje na kolo detektora vrha kako bi se ekstraktovala informacija amplitude primljenih paketa. U *feedback* konfiguraciji, vraćanje signala amplitude vrši se

u predpojačavačkoj fazi. Kako bi se formirala povratna sprega koristi se diferencijalni ulazno/izlazni transimpendansni pojačavač sa kolom detektora vrha. Na slici 3. prikazane su implementacije optičkih prijemnika.



Slika 3. *„Feedforward" i „feedback" implementacije optičkih prijemnika koji koriste mod praska* [5]

5. PRIMOPREDAJNIČKI DIGITALNO - DIJAGNOSTIČKI MONITORING

Primopredajnička digitalna dijagnostika nadgleda temperaturu modula, snagu prijemnika, prednaponsku struju predajnika i snagu predajnika. Izvan ovih parametara modula, postoji potreba da se nadgledaju greške u linku vlakna.

5.1. Nadgledanje modulskih parametara

Radne i dijagnostičke informacije nadgledaju se i prijavljuju digitalnim dijagnostičkim primopredajničkim kontrolerom (DDTC – *Digital Diagnostics Transceiver Controller*). Izlaz fizičke vrednosti svakog parametra je analogni napon ili struja od transimpendansnog pojačavača, laserskog upravljača ili post-pojačavača.

5.2. Nadgledanje temperature

Zahteva se precizno nadgledanje apsolutnih temperatura u okolinama bez aktivnih temperaturno-kontrolnih šema, sa velikim dinamičkim opsegom i boljom sveukupnom varijacijom od 3°C. Koji god senzori da se odaberu, odlučujući faktori su preciznost, stabilnost i ponovljivost opaženih rezultata.

5.3. Nadgledanje snage prijemnika

Zahteva se da preciznost nadgledanja snage prijemnika bude unutar $\pm$3dB. Najvažniji faktori, pored brojnih koji utiču na preciznost nadgledanja optičke snage prijemnika, su varijacija reagovanja fotodetektora u odnosu na temperaturu, napon i starenje.

5.4. Nadgledanje prednaponske struje predajnika

Zahtev za nadgledanje prednaponske struje predajnika je da preciznost bude bolja od $\pm$10%. Ovo je uobičajeno preciznije od temperaturnog nadgledanja i nadgledanja snage prijemnika, zato što ne pokriva toliko širok dinamičan opseg.

5.5. Nadgledanje snage predajnika

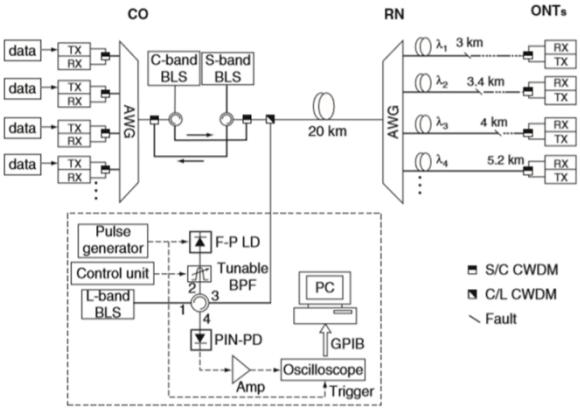
Zahtev za nadgledanje snage predajnika je $\pm$1dB. Napon nadgledane snage predajnika uobičajeno je u opsegu od nekoliko stotina milivolti do nekoliko volti.

5.6. Nadgledanje OTDR vlakna

Nadgledanje vlakna obavlja se merama OTDR-a na strani OLT-a na nametljiv način. Podaci OLT OTDR-a trpe smanjenje osetljivosti zbog velikih gubitaka splitovanja i dvosmislenosti zbog superpozicije tragova OTDR-a koji potiču od različitih grana vlakana.

5.7. Principi rada OTDR-a

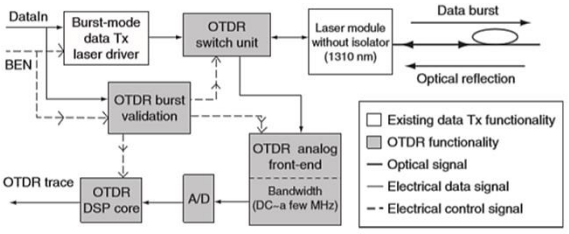
Prikazan je OTDR sistem u skladu sa tipičnom WDM-PON arhitekturom koja je bazirana na DFB LD-ovima ili na FP LD-ovima sa zaključanim talasnim dužinama (primer dat na slici 4.). Sastoji od širokopojasnog izvora svetlosti (BLS – *Broadband Light Source*), podesivog filtera za propusni opseg (BPF - *Broadband Bandpass Filter*), F-P LD-a, optičkog cirkulatora, OTDR prijemnika i drugih elektronskih delova.



Slika 4. *Primer podesivog OTDR-a za „in-service" nadgledanje greške vlakna u WDM-PON-u* [6]

5.8. OTDR koji je ugrađen u ONU

Integrisanjem OTDR funkcionalnosti u optičke primopredajničke module, ugrađeni OTDR postaje integralni deo mreže kome se može pristupiti sistemom upravljanja PON-a kako bi se nadgledao i testirao kvalitet fizičkog sloja. Blok dijagram ONU predajnika koji koristi *burst* mod (BM-Tx) sa ugrađenom neometajućom funkcionalnošću OTDR-a, sadrži saobraćaj podataka ulaza (DataIn), *burst* signala kome je omogućena transmisija (BEN – *Burst Enable*) i trag izlaza OTDR-a što se vidi na slici 6. OTDR metod može dati dobru preciznost za lociranje naglih promena u slabljenju kao klasični impulsni odziv povratnog rasipanja.



Slika 6. *Ugrađeni OTDR na ONU predajnik koji koristi burst mod. OTDR analogni prednji propusni opseg je ograničen na 5 MHz* [7]

6. PROCENA SISTEMA PRIMOPREDAJNIKA PON-a

Testiranja PON primopredajničkih sistema moraju uzeti u obzir prirodu *burst* podataka i disparitet u daljinama od svakog ONU-a do OLT-a. Tokom procesa OLT karakterizacije prijemne osetljivosti, amplituda signala i faza sata mogu se razlikovati od paketa do paketa. Stoga se uzastopni paketi moraju uzeti u obzir sa velikom razlikom u optičkoj snazi i poravnjanju faze sata kako bi se uhvatio najgori mogući scenario. Osetljivost OLT-a može se proceniti u tri scenarija: merenje kontinualnog moda, merenje jednog *burst* moda ONU-a i merenje dualnog *burst* moda ONU-a. Najvažniji parametri u proceni prijemnika *burst* moda su osetljivost, dinamički opseg i vreme reagovanja. Dinamički opseg sistema definisan je kao odnos najjačih i najslabijih optičkih signala koji mogu da se prime OLT prijemnikom koji koristi burst mod sa garantovanim specifikacijama performansi.

6.1. Procena sistema GPON prijemnika

GPON koristi implementaciju mehanizma izjednačavanja snage (PLM) kako bi se dobila specifikacija dinamičkog opsega od 21 dB OLT prijemnika. PLM dozvoljava ONU-ovima da rade u tri diskretna moda izlazne snage. PLM radi na sledeći način: OLT Rx prvo meri primljenu srednju snagu i upoređuje je sa dva praga napona. Onda odlučuje da li je dolazni optički signal previše nizak ili previše visok ili unutar opsega. Kada ONU primi poruku od OLT-a da promeni jedan mod u drugi, on postavlja svoju emitovanu snagu unutar opsega novog moda i onda nastavlja sa slanjem *upstream* podataka. Jedna prednost primenjivanja PLM-a je da se smanji zahtev dinamičkog opsega za 5-6 dB (od 21 dB do 15 dB) za OLT prijemnik. Još jedna prednost jeste da on povećava životni vek lasera i smanjuje potrošnju snage ONU-a kada radi u Modu 1 i/ili Modu 2.

6.2. EPON primopredajnici

EPON specifikacija dozvoljava naizmenično spojene OLT primopredajnike sa *burst* modom radi jednostavnosti i ekonomskih razloga. Postavke merenja burst moda prijemnika koji koristi dva ONU-a, češće su korišćene kako bi karakterisali EPON primopredajnike.

6.3. Uticaj analognog CATV "*overlay*"-a

PON primopredajnici mogu da podrže RF video „*overlay*" korišćenjem dodatne talasne dužine od 1.550 nm u *downstream*-u. Ovo ističe parametre optičke izolacije između analognih i digitalnih *downstream* talasnih dužina na ONU prijemniku i odlučuje kvalitet zahteva ONT WDM filtera.

7. ZAKLJUČAK

PON primopredajnici su jedinstveni bidirekcionalni uređaji koji koriste različite talasne dužine kako bi preneli i primili signale između OLT-a i ONU-ova na jednom vlaknu. Različiti tipovi PON-a razlikuju se po bitskim brzinama *upstream* i *downstream* transmisije, dometa, odnosa deljenja i dinamike burst moda, koja utiče na specifikacije odgovarajućeg primopredajnika. Jedan od kritičnih problema za PON primopredajnike jeste odnos performansi i troškova koji je meren tehničkim specifikacijama i jediničnom cenom optičkih primopredajnika. Glavni tehnološki izazovi u PON primopredajnicima su: (1) *upstream* tehnologije optičkog prenosa burst moda; (2) izlazna snaga optičkog predajnika kao i osetljivost optičkog prijemnika kako bi se zadovoljili zahtevi povećanja budžeta sistemskog linka; i (3) više brzine podataka. Iako su 1 Gbit/s PON sistemi kao što su EPON i GPON dobili na popularnosti, mnogo brži PON sistemi se zahtevaju u regijama gde mnogim biznis korisnicima trebaju linije većeg kapaciteta. Nedavno, IEEE 802.3 je započeo 802.3av radnu grupu koja aktivno razvija 10-Gbit/s EPON standarde. Ciljevi 10-G EPON radne grupe je da razviju simetrične i asimetrične 10-Gbit/s operacije linijske brzine gde će simetralna operacija raditi na 10 Gbit/s i u *downstream* i u *upstream* pravcu.

8. LITERATURA

[1] G. Kramer, „What is Next for Ethernet PON?" The 5th International Conference on Optical Internet (COIN 2006); MoB2–1.

[2] W. Huang, X. Li, C. Xu, X. Hong, C. Xu, and W. Liang, „Optical transceivers for fiber-tothe-premises applications: System requirements and enabling technologies," J. Lightwave Technol. vol.25, pp11–27, 2007.

[3] X.Z. Qiu, P. Ossieur, J. Bauwelinck, Y.C. Yi, D. Verhulst, J. Vandewege, B. De Vos, and P. Solina, „Development of G-PON upstream physical media dependent prototypes," J. Lightwave Technol., vol.22, pp2498–2508, Nov., 2004.

[4] Y. Chang and G. Noh, „1.25 Gb/s uplink burst–mode transmissions: System requirements and optical diagnostic challenges of EPON physical-layer chipset for enabling broadband optical Ethernet access networks," OFC/NFOEC'06 Paper JThB84.

[5] C.A. Eldering, „Theoretical determination of sensitivity penalty of burst-mode fiber optic receiver," J. Lightwave Technol., vol.11, pp2145–2149, Dec., 1993.

[6] J. Park, J. Baik, and C. Lee, „Fault-detection technique in a WDM-PON," Opt. Exp., vol.15, pp1461–1466, 2007.

[7] W. Chen, B.D. Mulder, J. Vandewege, X.Z. Qiu, J. Bauwelinck, and B. Baekelandt, „A novel technique for low-cost embedded non-intrusive fiber monitoring of P2MP optical access networks," OFC'07, Paper OThE4.

Kratka biografija:

Marija Avramović rođena je u Smederevu 1984. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Telekomunikacije odbranila je 2019. god.

kontakt: avramovic.marija@gmail.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.

Зборник радова Факултета техничких наука, Нови Сад

UDK: 621.3.049
DOI: https://doi.org/10.24867/07BE01Zoranovic

СОФТВЕРСКА ИМПЛЕМЕНТАЦИЈА МЕМФРАКТОРА – ГЕНЕРАЛИЗОВАНОГ ЕЛЕКТРИЧНОГ ЕЛЕМЕНТА СА МЕМОРИЈОМ

SOFTWARE IMPLEMENTATION OF MEMFRACTOR – GENERALIZED ELECTRICAL ELEMENT WITH MEMORY

Јована Зорановић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *У овом раду је описана софтверска имплементација математичког модела мемфракционог елемената. Дат је модел генерализованог елемента електричног кола, који је базиран на уопштеном Омовом закону и описан коришћењем фракционог рачуна. Фрактори, или општије мемфрактори, имплементирани су у MATLAB-у и Simulink-у. Интерполиране карактеристике мемелемената који су између мемкондензатора, мемристора, мемкалема и мемристора другог реда, приказане су као посебни случајеви карактеристика мемфрактора.*

Кључне речи: *мемфрактор, фрактор, фракциони рачун, мемристор, мемкондензатор, мемкалем*

Abstract – *In this paper software implementation of the mathematical model of memfractional element has been developed. The model of generalized electrical circuit element based on generalized Ohm's law is described using fractional calculus. Fractors, or more general memfractors, are implemented in MATLAB and Simulink. The interpolated characteristics of memelements between a memcapacitor, memristor, meminductor and 2nd-order memristor, are shown as particular cases of memfractor's characteristics.*

Keywords: *memfractor, fractor, fractional calculus, memristor, memcapacitor, meminductor*

1. УВОД

Постојање четвртог основног елемента кола – мемристора – је први пут описано 1971. у раду Леона Чуе [1]. Осим мемристора, меморијске особине се могу уочити и код других двокрајних елемената кола. Ти елементи су мемкондензатори, мемкалемови и мемристори другог реда, као и елементи вишег реда, обухваћени термином мемфрактор [2]. За мемфрактор се може рећи да је то сваки елемент кола који испољава следеће две особине: а) поседовање меморије и б) присуство нецелобројних извода или интеграла у својим конститутивним релацијама. Остатак текста овог рада организован је на следећи начин: У другом поглављу дата је математичка основа фракционог рачуна, док су у трећем поглављу дефинисани мемристор, мемкалем и мемкондензатор, као и уопштен Омов закон и мемфрактор. У четвртом поглављу описана је софтверска имплементација фракционог рачуна, која је приказана на примерима различитих мемфрактора. Примена *toolbox*-а и библиотеке која ради са фракционим рачуном у *MATLAB*-у и *Simulink*-у, као и формирање модела мемфрактора описана је у петом поглављу. У претпоследњем и последњем поглављу дат је закључак и списак коришћене литературе.

НАПОМЕНА:
Овај рад је проистекао из мастер рада чији ментор је био др Станиша Даутовић, доцент.

2. ФРАКЦИОНИ РАЧУН – ИСТОРИЈА И ДЕФИНИЦИЈА

Интегрални и диференцијални рачун представљају основне градивне елементе данашње математике и технологије и њихово познавање је неопходно да би се моделовали природни и вештачки системи. Фракциони (нецелобројни) рачун [3] је уопштење традиционалне дефиниције интегралног и диференцијалног рачуна, уз додатно проширење нецелобројним интегрaљењем и диференцирањем. Фракциони интеграл дефинисан је уопштењем Кошијевог интеграла за реалне позитивне вредности параметра α, и дат је изразом [4]:

$$D^{-\alpha}f(t) \coloneqq \frac{1}{\Gamma(\alpha)}\int_0^t (t-\tau)^{\alpha-1}f(\tau)d\tau \text{ за } \alpha \in \mathbb{R}^+. \quad (1)$$

Најчешће коришћена дефиниција фракционог диференцијала јесте Риман-Лиувил-ова (*Riemann-Liouville*) и дата је изразом [4]

$$D^{\alpha}f(t) \coloneqq \begin{cases} \frac{d^m}{dt^m}\left[\frac{1}{\Gamma(m-\alpha)}\int_0^t \frac{f(\tau)}{(t-\tau)^{\alpha+1-m}}\right], & m-1<\alpha<m, \\ \frac{d^m}{dt^m}f(t), & \alpha = m\,. \end{cases} \quad (2)$$

где $\alpha \in \mathbb{R}^+$, $m \in \mathbb{N}$, са ограничењем $m-1 < \alpha \leq m$ [4].

У изразима (1) и (2), $\Gamma(z)$ представља гама функцију (eng. *Gamma function*) односно уопштење факторијела за реалне вредности. Гама функција је дефинисана изразом [4]

$$\Gamma(z) = \int_0^\infty e^{-u}u^{z-1}du \text{ за } z \in \mathbb{R}, \quad (3)$$

док су њене често коришћене особине

$$\Gamma(z+1) = z\Gamma(z), \quad (4)$$

$$\text{када је } z \in \mathbb{N} \Rightarrow \Gamma(z) = (z-1)!\,. \quad (5)$$

У овом раду коришћена је нумеричка метода за рачунање фракционог интеграла и диференцијала, која се назива Грунвалд-Летникова (*Grunwald-Letnikov*) метода. За $\alpha \in \mathbb{R}^+$, Грунвалд-Летников фракциони интеграл дефинисан је изразом [2]:

$$D^{-\alpha}f(t) = \lim_{h\to 0} h^{\alpha} \sum_{m=0}^{\frac{t-a}{h}} \frac{\Gamma(\alpha+m)}{m!\Gamma(\alpha)} f(t-mh), \quad (6)$$

док је фракциони извод дефинисан са

$$D^{\alpha}f(t) = \lim_{h\to 0} \frac{1}{h^{\alpha}} \sum_{m=0}^{\frac{t-a}{h}} (-1)^m \frac{\Gamma(\alpha+1)}{m!\Gamma(\alpha-m+1)} f(t-mh). \quad (7)$$

Када је α=0,

$$D^{0}f(t) = f(t). \quad (8)$$

3. МЕМРИСТОР, МЕМКОНДЕНЗАТОР, МЕМКАЛЕМ И МЕМФРАКТОР

У теорији електричних кола, фундаменталне променљиве су струја i и напон u. Овим величинама се придружују и количина наелектрисања q и флукс φ, где се под количином наелектрисања q уобичајено подразумева интеграл струје, $q(t) = \int_{-\infty}^{t} i(\tau)d\tau$, а под флуксом φ интеграл напона, $\varphi(t) = \int_{-\infty}^{t} u(\tau)d\tau$.

Отпорник је дефинисан у $u-i$ равни конститутивном релацијом (кратко КР) облика $f(u,i)=0$. Аналогно, калем је дефинисан својом КР у $\varphi - i$ равни, $f(\varphi,i)=0$, и коначно кондензатор је дефинисан својом КР у $q-u$ равни, $f(q,u)=0$. Поред наведених дефиниционих израза, једина веза која недостаје је зависност количине наелектрисања и флукса. Конститутивна релација облика $f(\varphi,q)=0$ дефинише тзв. идеални мемристор, где је f нелинеарна функција (за f која је линеарна функција, мемристор дегенерише у отпорник). Алтернативна дефиниција мемристора је да је то било који двокрајни уређај који манифестује уштинуту хистерезисну петљу, која увек пролази кроз координатни почетак у напонско-струјној равни, када се на улаз доведе периодичан непарно симетричан сигнал са нултом једносмерном компонетом, било да је на улазу независтан струјни или напонски генератор [5]. Уколико је улаз струја, мемристор се назива струјом контролисан, а уколико се на улаз доведе напон, мемристор је напоном контролисан.

Временски непроменљив мемристор се такође може дефинисати преко тзв. уопштеног Омовог закона. На овај начин, струјом контролисан мемристор је дефинисан изразом [6]

$$\left.\begin{aligned} u &= M(x_1,x_2,\ldots,x_n)\, i \\ \frac{dx_k}{dt} &= f_k(x_1,x_2,\ldots,x_n;\, i), \quad k=1,2,\ldots,n \end{aligned}\right\} \quad (9)$$

где је M континуална функција n променљивих стања $x_1, x_2, \ldots, x_n$, названа мемристанса (eng. *memristance*).

Дуално, напоном контролисан мемристор је дефинисан изразом [6]

$$\left.\begin{aligned} i &= W(x_1,x_2,\ldots,x_n)\, u \\ \frac{dx_k}{dt} &= f_k(x_1,x_2,\ldots,x_n;\, v), \quad k=1,2,\ldots,n \end{aligned}\right\} \quad (10)$$

где је W континуална функција n променљивих стања $x_1, x_2, \ldots, x_n$, названа мемдуктанса (eng. *memductance*).

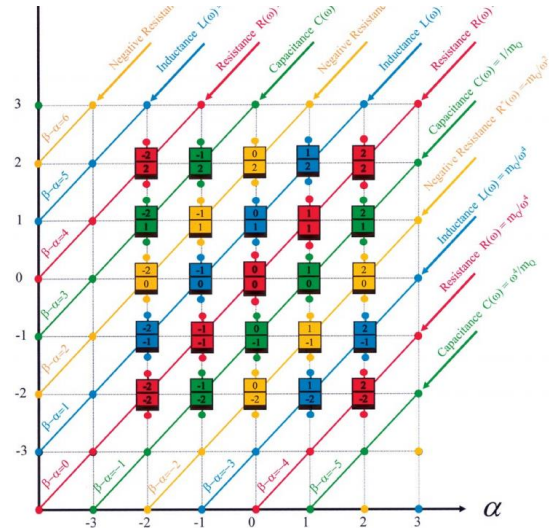
Уопштени Омов закон који збирно обухвата све класе елемената кола дат је изразом [2]

$$u(t) = D^{-1-\alpha}(M^{\alpha,\beta} D^{\beta+2}(q(t))), \quad (11)$$

где је $M^{\alpha,\beta}$ једнако

$$M^{\alpha,\beta} = \begin{cases} R, & \text{за } \alpha=\beta=0, \text{отпорник}, \\ C^{-1}, & \text{за } \alpha=0, \beta=-1, \text{ кондензатор}, \\ L, & \text{за } \alpha=-1, \beta=0, \text{ калем}, \\ R_{\text{M}}, & \text{за } \alpha=\beta=-1, \text{мемристор}, \\ C_M^{-1}, & \text{за } \alpha=-1, \beta=-2, \text{ мемкондензатор}, \\ L_{\text{M}}, & \text{за } \alpha=-2, \beta=-1, \text{ мемкалем}, \\ R_{2\text{M}}, & \text{за } \alpha=\beta=-2, \text{мемр. другог реда}, \\ C_{2M}^{-1}, & \text{за } \alpha=-2, \ \beta=-3, \text{ мемкон. др. реда}, \\ L_{2\text{M}}, & \text{за } \alpha=-3, \ \beta=-2, \text{мемкалем др. реда}, \\ R_{3\text{M}}, & \text{за } \alpha=\beta=-3, \text{ мемристор трећег реда}. \end{cases} \quad (12)$$

Као што се са слике 3.1 може видети, у пресеку хоризонталних и вертикалних правих које пролазе кроз целобројне вредности параметара α и β, налазе се елементи који припадају једној од 4 класа електричних елемената (фреквенцијски зависни позитивни и негативни отпорници, калеми и кондензатори). Сви елементи чија је вредност β-α једнака, припадају истој класи. Анализом особина елемената за различите вредности β-α, примећена је периодичност по модулу ± 4 и по параметру α и по β, као што је то приказано на слици 3.1. Због ове особине периодичности, слика 3.1 је у [7] названа периодни систем свих двокрајних елемената кола, по угледу на периодни систем хемијских елемената.



Слика 3.1. *Периодни систем двокрајних елемената кола* (преузето из [7]).

Коришћењем нецелобројног фракционог рачуна могуће је дефинисати елементе кола и за реалне вредности параметара α и β, што проширује периодни систем елемената са важења над дводимензионалном мрежом $\mathbb{Z}^2$ на целу $\mathbb{R}^2$ раван. Без фракционог рачуна, елементи су били дефинисани само у чворовима 2Д целобројне мреже $\mathbb{Z}^2$.

4. СОФТВЕРСКА ИМПЛЕМЕНТАЦИЈА МАТЕМАТИЧКОГ МОДЕЛА МЕМФРАКТОРА

На следећим примерима је приказан резултат рада софтверске имплементације мемфрактора у *MATLAB*-у. Пример је преузет из [1] и реимплементиран у *MATLAB*-у.

Нека је мемкондензатор задат функцијом

$$C_M(\sigma) = \frac{1}{1+(2+\sigma)^2} \quad (13)$$

где је σ интеграл количине наелектрисања q ($\sigma(t) = \int_{-\infty}^{t} q(\tau)d\tau$). Надаље, нека је мемристор задат својом мемристансом

$$R_M(q) = 1 + q + q^2, \quad (14)$$

мемкалем функцијом

$$L_M(q) = 1 + e^q, \quad (15)$$

и мемристор другог реда мемристансом другог реда

$$R_{2M}(\sigma) = 1 + \sigma + \sigma^2. \quad (16)$$

Ова четири елемента су сваки за себе повезани са независним струјним генератором чији је сигнал једнак

$$i(t) = \begin{cases} \cos(t), & t \geq 0, \\ 0, & t < 0. \end{cases} \quad (17)$$

Под претпоставком да је почетни услов q_0=q(0)=0

$$q(t) = \int_0^t i(\tau)d\tau = \sin(t), \quad (18)$$

тада је

$$\sigma(t) = \int_0^t q(\tau)d\tau = 1 - \cos(t). \quad (19)$$

Заменом параметара α и β и функције $M^{\alpha,\beta}$ у уопштеном Омовом закону датог изразом (11), и то $\alpha \to \alpha_1 - 2$, $\beta \to \alpha_2 - 2$ и $M^{\alpha,\beta} \to F_M^{\alpha_1,\alpha_2}(t)$ добија се

$$u(t) = D^{1-\alpha_1}\left(F_M^{\alpha_1,\alpha_2}(t)D^{\alpha_2}\left(q(t)\right)\right), \quad (20)$$

то јест у нашем примеру

$$u(t) = D^{1-\alpha_1}\left(F_M^{\alpha_1,\alpha_2}(t)D^{\alpha_2}\left(\sin(t)\right)\right), \quad (21)$$

где је

$$F_M^{\alpha_1,\alpha_2} = a_{(\alpha_1,\alpha_2)}C_M^{-1}(t) + b_{(\alpha_1,\alpha_2)}R_M(t) + c_{(\alpha_1,\alpha_2)}L_M(t) + d_{(\alpha_1,\alpha_2)}R_{2M}(t), \quad (22)$$

уз вредност коефицијената

$$a_{(\alpha_1,\alpha_2)} = \alpha_1(1-\alpha_2), \quad (23)$$

$$b_{(\alpha_1,\alpha_2)} = \alpha_1\alpha_2\left(\frac{\alpha_1+\alpha_2}{2}\right), \quad (24)$$

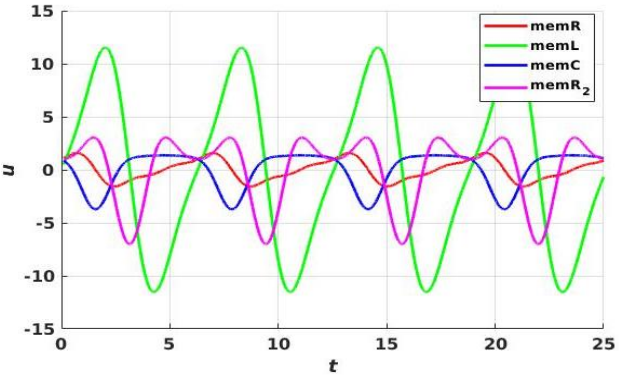
$$c_{(\alpha_1,\alpha_2)} = \alpha_2(1-\alpha_1), \quad (25)$$

$$d_{(\alpha_1,\alpha_2)} = (1-\alpha_1)(1-\alpha_2). \quad (26)$$

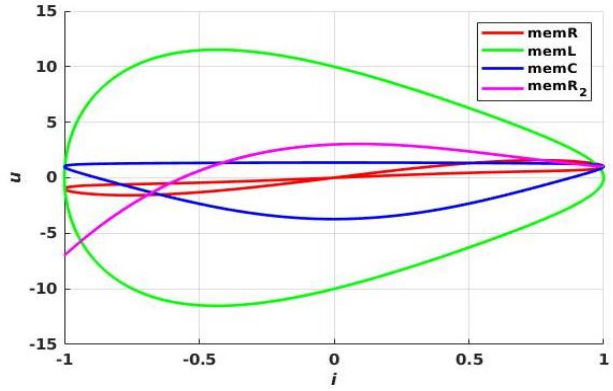
За вредности $\alpha_1 = 1$ и $\alpha_2 = 1$ реч је о мемристору, за $\alpha_1 = 0$ и $\alpha_2 = 1$ о мемкалему, за $\alpha_1 = 1$ и $\alpha_2 = 0$ о мемкондензатору и за $\alpha_1 = 0$ и $\alpha_2 = 0$ о мемристору другог реда. Ови закључци следе на основу једноставних веза између параметара (α_1, α_2) и (α, β), као и упоређивањем слика 3.1. и 4.1.

Слика 4.1: *Шематски приказ односа мемристора, мемкондензатора, мемкалема и мемристора другог реда у α_1-α_2 равни.*

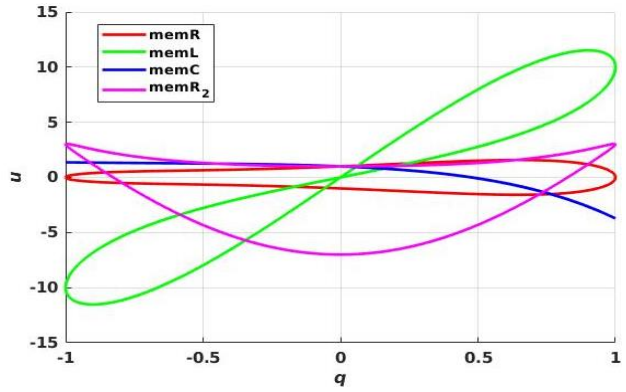
На сликама 4.2, 4.3 и 4.4 су редом приказане функције $u(t)$, $u(i)$ и $u(q)$. На овим графицима крива црвене боје описује мемристор, крива зелене боје мемкалем, крива плаве боје мемкондензатор и крива розе боје мемристор другог реда.



Слика 4.2. *Приказ таласног облика напона.*

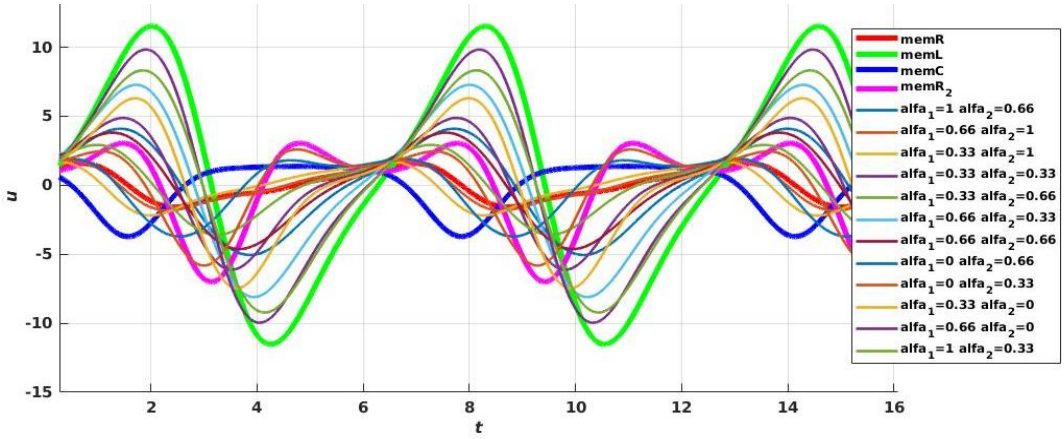


Слика 4.3. *Струјно-напонска к-ка за мемристор, мемкалем, мемконд. и мемристор другог реда.*

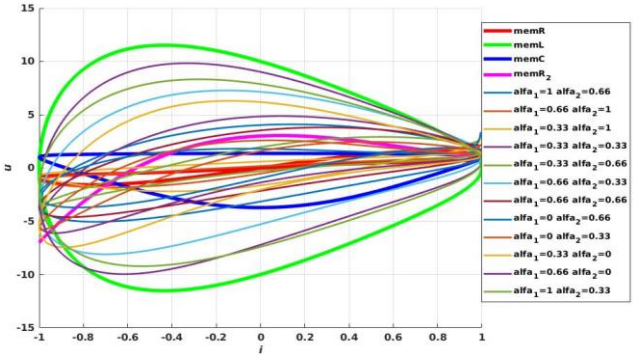


Слика 4.4. *Приказ $u(q)$ за мемристор, мемкалем, мемкондензатор и мемристор другог реда.*

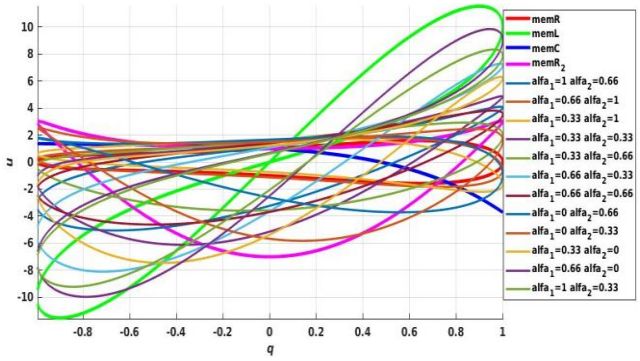
На сликама 4.5, 4.6 и 4.7 су редом приказане функције $u(t)$, $u(i)$ и $u(q)$. На овим графицима су поред мемристора, мемкалема, мемкондензатора и мемристора другог реда, приказане карактеристике елемената који су добијени „хибридизацијом“ ова четири основна елемента, у смислу да су карактеристике нових елемената „интерполиране“ негде између карактеристика четири елемента са слике 4.1. Код њих су α_1 и α_2 нецелобројне вредности $0 \leq \alpha_1 \leq 1$ и $0 \leq \alpha_2 \leq 1$. На сликама 4.5, 4.6 и 4.7 су приказане карактеристике мемфрактанси које одговарају свим преосталим комбинацијама два елемента између четири вредности из скупа {0, 0.33, 0.66, 1}.



Слика 4.5. *Приказ u(t) за различите вредности параметара,* (α_1, α_2) $0 \leq \alpha_1 \leq 1$ *и* $0 \leq \alpha_2 \leq 1$.



Слика 4.6. *Приказ u*(i) *за* $0 \leq \alpha_1 \leq 1$ *и* $0 \leq \alpha_2 \leq 1$.



Слика 4.7. *Приказ u*(q) *за* $0 \leq \alpha_1 \leq 1$ *и* $0 \leq \alpha_2 \leq 1$.

5. MATLAB TOOLBOX ЗА ФРАКЦИОНИ РАЧУН

Toolbox који је коришћен у овом раду носи назив *FOMCON* [8]. Као и већина *toolbox*-ова који раде са фракционим рачуном, тако је и код овог случај да су развијени за област аутоматике. У овом раду је дати *toolbox* модификован за потребе области електронике, те је проширен основни сет функција и *Simulink* модел. Поновљен је пример из поглавља 4 и у самом *MATLAB* делу као и у *Simulink*-у.

6. ЗАКЉУЧАК

У овом раду описана је софтверска имплементација мемфрактансних електричних елемената, чије су карактеристике базиране на нецелобројним изводима и интегралима, као и на проширењу скупа основних електричних елемената (отпорник, калем, кондензатор) новим елементима са меморијом (мемристор, мемкондензатор и мемкалем првог и виших целобројних редова). На тај начин је омогућен опис математичког модела и алгоритамска симулација понашања елемената у било којој тачки унутар $\alpha - \beta$ равни (приказане на слици 3.1). Ови елементи до сада нису били имплементирани у *MATLAB*-у и *Simulink*-у, као ни у специјализованим наменским софтверским библиотекама за фракциони рачун, као што је *FOMCON*.

7. ЛИТЕРАТУРА

[1] L. Chua, "Memristor - The Missing Circuit Element", *IEEE Transactions on Circuit Theory*, vol. 18, no. 5, 1971.

[2] M.-S. Abdelouahab, R. Lozi and L. Chua, "Memfractance: A Mathematical Paradigm for Circuit Elements with Memory", *International Journal of Bifurcation and Chaos, World Scientific Publishing*, pp.1430023- 1430023-29, 2014.

[3] I. Podlubny, "Fractional differential equations", *Academic Press, San Diego*, 1999.

[4] A. Loverro, "Fractional Calculus History, Definitions and Applications for the Engineer", *Rapport technique, Univeristy of Notre Dame*, IN 46556, U.S.A., 2004.

[5] L.Chua, "Everything You Wish to Know About Memristors but are afraid to ask", *Radioengineering*, vol. 24, no. 2, 2015.

[6] Shyam Prasad Adhikari, et al, "Three fingerprints of Memristor", *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 11, 2013.

[7] L. Chua et al, "Nonlinear Circuit Foundations for Nanodevices", *Proceedings of the IEEE*, vol. 91, no. 11, 2003.

[8] A. Tepljakov, E. Petlenkov, and J. Belikov, "Fomcon: a *MATLAB* Toolbox for Fractional- order System Identification and Control", *International journal of microelectronics and computer science*, vol. 2, no. 2, 2011.

Кратка биографија:

Јована Зорановић рођена је у Новом Саду 1994. године. Факултет техничких наука је уписала 2013. године, а дипломски B.Sc. рад одбранила 2018. године.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/07BE02Pavlovic

UTICAJ STRUJNIH MERENJA NA PRORAČUN STATIČKE ESTIMACIJE STANJA U PRENOSNIM MREŽAMA

INFLUENCE OF CURRENT MEASUREMENTS ON STATIC STATE ESTIMATION CALCULATION IN ELECTRIC POWER SYSTEMS

Miroslav Pavlović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu razmatrana je statička estimacija stanja, kao bazična funkcija za većinu proračuna vezanih za analizu, eksploataciju i upravljanje elektroenergetskim sistemima. Posebna pažnja posvećena je korišćenju strujnih merenja u proračunu statičke estimacije. Predstavljen je matematički model korišćen za proračun statičke estimacije stanja u prisustvu strujnih merenja i izvršena verifikacija modela na test sistemima sa 3 i 14 čvorova.*

Ključne reči: *Statička estimacija stanja, observabilnost sistema, strujna merenja.*

Abstract – *In this paper, the static state estimation is considered, as a basic function for most of the calculations related to the analysis, exploitation and management of the electric power system. Special attention is paid to the use of current measurements in the static state estimation calculation. Mathematical model used for static state estimation calculation, in a presence of current measurements, is introduced and model verification is performed on 3 and 14-bus test systems.*

Keywords: *Static state estimation, system observability, current measurements.*

1. UVOD

Statička estimacija stanja predstavlja bazičnu funkciju za većinu proračuna vezanih za analizu, eksploataciju i upravljanje elektroenergetskim sistemima (EES).

Kompletan proces statičke estimacije stanja odvija se kroz 6 koraka [1]: Formiranje modela, analiza observabilnosti sistema, statička estimacija stanja, detekcija i identifikacija loših merenja i estimacija parametara modela mreže.

Cilj ovog rada jeste da se da teorijska postavka prva 3 koraka statičke estimacije stanja.

Posebna pažnja biće posvećena strujnim merenjima koja su danas široko rasprostranjena u svim EES-ima. Biće objašnjene prednosti, kao i nedostaci korišćenja strujnih merenja u proračunu statičke estimacije stanja, odnosno metodu minimuma sume otežanih kvadrata reziduala.

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

Rad će pored jasne postavke problema ponuditi i predloge za rešavanje istih, u skladu sa rezultatima algoritma implementiranog u programskim jezicima C++ i Fortran. U okviru primera, na test sistemima, biće izvršena analiza osetljivosti estimiranih vrednosti napona na grešku merenja modula struje u grani, kao i osetljivost estimiranih vrednosti na promenu izmerene vrednosti modula napona u čvorovima mreže.

2. POSTAVKA PROBLEMA

Osnovni pojmovi potrebni za razumevanje algoritma statičke estimacije stanja su: vektor stanja $\boldsymbol{x}$, vektor merenja $\boldsymbol{z}$, varijansa σ_m^2, težinski faktor w_m i redundansa merenja R_d.

Vektor stanja predstavlja n-dimenzioni vektor promenljivih stanja $\boldsymbol{x}$, izraz (1).

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{V} & \boldsymbol{\theta} & \boldsymbol{a_l} & \boldsymbol{\varphi_l} \end{bmatrix}^T , \tag{1}$$

gde je $\boldsymbol{V}$ vektor modula fazora napona, $\boldsymbol{\theta}$ vektor uglova fazora napona u svim čvorovima (osim referentnog balansnog), a $\boldsymbol{a_l}$ i $\boldsymbol{\varphi_l}$ vektori nenominalnih odnosa transformacije klasičnih i faznih regulacionih transformatora, respektivno.

Vektor merenja $\boldsymbol{z}$ povezan je sa vektorom promenljivih stanja $\boldsymbol{x}$ preko nelinearne jednačine:

$$\boldsymbol{z} = \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{e} \ , \tag{2}$$

gde je $\boldsymbol{z}$ M-dimenzioni vektor merenja, $\boldsymbol{h(x)}$ M-dimenzioni vektor funkcija promenljivih stanja, $\boldsymbol{x}$ n-dimenzioni vektor promenljivih stanja i $\boldsymbol{e}$ M-dimenzioni slučajni vektor grešaka merenja.

Vektor merenja koji se koristi u statičkoj estimaciji stanja se može predstaviti na sledeći način (preko 9 subvektora):

$$\boldsymbol{z} = \begin{bmatrix} \boldsymbol{P_{ij}^T} & \boldsymbol{Q_{ij}^T} & \boldsymbol{I_{ij}^T} & \boldsymbol{P_i^T} & \boldsymbol{Q_i^T} & \boldsymbol{I_i^T} & \boldsymbol{V_i^T} & \boldsymbol{a_l^T} & \boldsymbol{\varphi_l^T} \end{bmatrix}^T \tag{3}$$

gde su $\boldsymbol{P_{ij}}$,$\boldsymbol{Q_{ij}}$ i $\boldsymbol{I_{ij}}$ vektori tokova aktivnih i reaktivnih snaga i struja po granama, $\boldsymbol{P_i}$,$\boldsymbol{Q_i}$ i $\boldsymbol{I_i}$ vektori injektiranih aktivnih i reaktivnih snaga i struja u čvorovima, $\boldsymbol{V_i}$ vektor modula napona u čvorovima i $\boldsymbol{a_l}$ i $\boldsymbol{\varphi_l}$ vektori nenominalnih odnosa transformacije klasičnih i faznih regulacionih transformatora, respektivno.

Matrica težinskih faktora ima sledeći oblik:

$$\boldsymbol{W} = diag\left\{1/\sigma_m^2\right\}, \tag{4}$$

gde je σ_m^2 varijansa *m*-tog merenja, koja daje indikaciju o kvalitetu merenja.

Redundansa merenja R_d definiše se kao odnos broja raspoloživih merenja M i minimalno potrebnog broja nezavisnih merenja, koji praktično odgovara broju promenljivih stanja sistema n, pri čemu je $M > n$:

$$R_d = \frac{M}{n} \ . \tag{5}$$

2.1 Observabilnost sistema

Pod pojmom observabilnosti sistema podrazumeva se mogućnost da se nad razmatranom mrežom izvrši statička estimacija stanja, odnosno da se odredi rešenje vektora promenljivih stanja sa raspoloživim skupom merenja [2].

U skladu sa tim može se izvršiti podela merenja na *kritična merenja* čijim eliminisanjem sistem postaje neobservabilan, *nekritična* (*redundantna*) *merenja* čijim uklanjanjem sistem ostaje observabilan (povećavaju redundansu merenja, a samim tim i tačnost proračuna) i *merenja kritična za jedinstvenost rešenja* čijom eliminacijom sistem postaje nejedinstveno observabilan.

2.2 Kriterijum srednje-kvadratnih odstupanja

Problem statičke estimacije moguće je formulisati na sledeći način: Pronaći vrednosti promenljivih stanja, pri kojima se ima minimalna suma kvadrata odstupanja između izmerenih i estimiranih vrednosti veličina, pri čemu se svakoj razlici (rezidualu) dodeljuje odgovarajući težinski faktor [3]:

$$\min_{\boldsymbol{x}} \left\{ J(\boldsymbol{x}) = \sum_{m=1}^{M} \frac{\left[z_m - h_m(\boldsymbol{x}) \right]^2}{\sigma_m^2} \right\} \ , \tag{6}$$

gde je $J(\boldsymbol{x})$ ukupna suma kvadrata reziduala merenja (kriterijumska funkcija koja se minimizira), a $h_m(\boldsymbol{x})$ oznaka za funkciju *m*-tog merenja.

Sam vektor reziduala se može predstaviti kao:

$$\boldsymbol{r} = \boldsymbol{z} - h(\boldsymbol{x}) \ , \tag{7}$$

gde je $\boldsymbol{r}$ vektor reziduala merenja, $\boldsymbol{z}$ vektor merenja i $h(\boldsymbol{x})$ vektor estimiranih vrednosti merenja za dati vektor stanja.

Problem (6) je pogodno izraziti u kompaktnoj formi:

$$\min_{\boldsymbol{x}} \left\{ \mathrm{J}(\boldsymbol{x}) = \boldsymbol{r}^T \boldsymbol{R}^{-1} \boldsymbol{r} = [\boldsymbol{z} - \boldsymbol{h}(\boldsymbol{x})]^T \boldsymbol{R}^{-1} [\boldsymbol{z} - \boldsymbol{h}(\boldsymbol{x})] \right\} , \tag{8}$$

gde je $\boldsymbol{R}^{-1}$ već definisana dijagonalna matrica težinskih faktora $\boldsymbol{W}$.

Ako se funkcija vektora merenja $\boldsymbol{h(x)}$, iz izraza (2) linearizuje, dobija se sledeća funkcija:

$$\boldsymbol{z} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{e} \ , \tag{9}$$

gde $\boldsymbol{H}$ predstavlja ($M \times n$) dimenzionu Jakobijan matricu, odnosno osetljivost funkcije vektora merenja $\boldsymbol{h(x)}$ na promenu vektora promenljivih stanja $\boldsymbol{x}$:

$$\boldsymbol{H} = \left. \frac{\partial \boldsymbol{h}(\boldsymbol{x})}{\partial \boldsymbol{x}} \right|_{\boldsymbol{x} = \hat{\boldsymbol{x}}} \ . \tag{10}$$

gde se elementi vektora $\boldsymbol{h(x)}$ proračunavaju različito, u zavisnosti od tipa merenja.

3. MATEMATIČKI MODEL

U ovom poglavlju razmatraće se način rešavanja postavljenog problema statičke estimacije stanja. Biće opisane matrice Jakobijana $\boldsymbol{H}$ i pojačanja $\boldsymbol{G}$, uz poseban osvrt na tretman strujnih merenja u proračunu statičke estimacije stanja.

3.1 Metod normalnih jednačina

Minimum optimizacione funkcije (8) dobija se izjednačavanjem prvog izvoda funkcije, po vektoru stanja, sa nulom:

$$\frac{\partial J(\boldsymbol{x})}{\partial \boldsymbol{x}} = \boldsymbol{H}(\boldsymbol{x})^T \boldsymbol{R}^{-1} \left[\boldsymbol{z} - \boldsymbol{h}(\boldsymbol{x}) \right] = 0 \ . \tag{11}$$

Estimacija vektora promenljivih stanja se dobija iterativnim rešavanjem prethodnog sistema nelinearnih jednačina, koji se u svakoj iteraciji linearizuje, čime se dobija sledeći sistem linearnih jednačina [4]:

$$\boldsymbol{G}(\boldsymbol{x}^{(k)}) \boldsymbol{\Delta x}^{(k)} = \boldsymbol{H}(\boldsymbol{x}^{(k)}) \boldsymbol{R}^{-1} \boldsymbol{\Delta z}^{(k)}; k = 0,1,2... \tag{12a}$$

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \Delta \boldsymbol{x}^{(k)} \ , \tag{12b}$$

gde je $\boldsymbol{G}(\boldsymbol{x}^{(k)}) = \boldsymbol{H}(\boldsymbol{x}^{(k)})^T \boldsymbol{R}^{-1} \boldsymbol{H}(\boldsymbol{x}^{(k)})$ matrica pojačanja u *k*-toj iteraciji, $\boldsymbol{\Delta z}^{(k)} = \boldsymbol{z} - h(\boldsymbol{x}^{(k)})$ priraštaj vektora merenja u *k*-toj iteraciji, $\boldsymbol{H}(\boldsymbol{x}^{(k)})$ matrica Jakobijana u *k*-toj iteraciji i $\boldsymbol{\Delta x}^{(k)}$ korekcija vektora stanja u *k*-toj iteraciji.

Iterativni postupak se završava kada je ispunjen sledeći uslov:

$$\max\{\boldsymbol{\Delta x}\} \le \xi \ , \tag{13}$$

gde je ξ unapred zadati kriterijum konvergencije.

3.2 Formiranje Jakobijan matrice

Dimenzije i struktura matrice Jakobijana zavise od vektora raspoloživih merenja i vektora promenljivih stanja [2]:

$$\boldsymbol{H} = \begin{bmatrix} \frac{\partial \boldsymbol{P}_{ij}}{\partial \boldsymbol{V}} & \frac{\partial \boldsymbol{Q}_{ij}}{\partial \boldsymbol{V}} & \frac{\partial \boldsymbol{I}_{ij}}{\partial \boldsymbol{V}} & \frac{\partial \boldsymbol{P}_{i}}{\partial \boldsymbol{V}} & \frac{\partial \boldsymbol{Q}_{i}}{\partial \boldsymbol{V}} & \frac{\partial \boldsymbol{I}_{i}}{\partial \boldsymbol{V}} & \frac{\partial \boldsymbol{V}_{i}}{\partial \boldsymbol{V}} \\ \frac{\partial \boldsymbol{P}_{ij}}{\partial \boldsymbol{\theta}} & \frac{\partial \boldsymbol{Q}_{ij}}{\partial \theta} & \frac{\partial \boldsymbol{I}_{ij}}{\partial V} & \frac{\partial \boldsymbol{P}_{i}}{\partial \boldsymbol{\theta}} & \frac{\partial \boldsymbol{Q}_{i}}{\partial \boldsymbol{\theta}} & \frac{\partial \boldsymbol{I}_{i}}{\partial \boldsymbol{\theta}} & \frac{\partial \boldsymbol{V}_{i}}{\partial \boldsymbol{\theta}} \end{bmatrix}^T . \tag{14}$$

Elementi matrice Jakobijana koji odgovaraju strujnim merenjima, definišu se na sledeći način:

$$\frac{\partial I_{ij}}{\partial V_i} = \frac{g_{ij}^2 + b_{ij}^2}{I_{ij}} \left(V_i - V_j \cos\theta_{ij} \right) \ , \tag{15a}$$

$$\frac{\partial I_{ij}}{\partial V_j} = \frac{g_{ij}^2 + b_{ij}^2}{I_{ij}} \left(V_j - V_i \cos\theta_{ij} \right) \ , \tag{15b}$$

$$\frac{\partial I_{ij}}{\partial \theta_i} = \frac{g_{ij}^2 + b_{ij}^2}{I_{ij}} V_i V_j \sin\theta_{ij} \ , \tag{15c}$$

$$\frac{\partial I_{ij}}{\partial \theta_j} = -\frac{g_{ij}^2 + b_{ij}^2}{I_{ij}} V_i V_j \sin\theta_{ij} \ . \tag{15d}$$

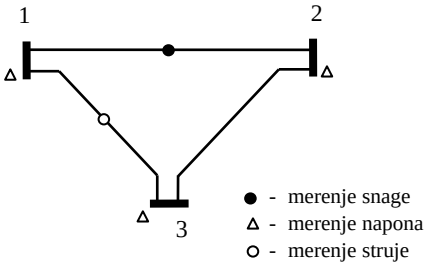
Ukoliko u sistemu postoje merenja injektiranih struja u čvorove, ona mogu biti iskorišćena u proračunu statičke estimacije stanja. Ovakva merenja uključuju se u proračun na veoma sličan način kao i merenja struja u granama i sa sobom nose iste probleme vezane za korišćenje strujnih merenja u statičkom estimatoru stanja [2].

4. VERIFIKACIJA MATEMATIČKOG MODELA

Verifikacija matematičkog modela za proračun statičke estimacije stanja izvršena je na 2 test sistema. Prvi test sistem je formiran od 3, a drugi od 14 čvorova [2].

4.1 Test mreža sa 3 čvora – sa strujnim merenjima

Test sistem sa 3 čvora prikazan je na slici 1.



Slika 1 – Test mreža sa 3 čvora

Parametri koji opisuju topologiju mreže dati su u relativnim jedinicama i prikazani u tabeli 1.

Tabela 1 – Podaci o topologiji sistema sa 3 čvora [2]

Čvor 1	Čvor 2	Tip	r[r.j.]	x[r.j.]	b[r.j.]	g[r.j.]
1	2	vod	0,01	0,03	0	0
1	3	vod	0,02	0,05	0	0
2	3	vod	0,03	0,08	0	0

Vrednosti i raspored merenja korišćenih u proračunu prikazani su u tabeli 2.

Tabela 2 – Podaci o merenjima sistema sa 3 čvora [2]

Čvor 1	Čvor 2	Tip	Vrednost merenja [r.j.]
2	2	P_2	–0,501
2	2	Q_2	–0,286
1	2	P_{12}	0,888
1	3	P_{13}	1,173
1	2	Q_{12}	0,568
1	3	Q_{13}	0,663
1	1	V_1	1,006
2	2	V_2	0,968

Ukoliko se za inicijalne vrednosti vektora stanja $\boldsymbol{x}$ usvoje nominalne vrednosti, elementi matrice Jakobijana koji odgovaraju strujnim merenjima jednaki su nuli. Iz ove činjenice može se zaključiti da se strujna merenja ne mogu iskoristiti u situaciji kada za inicijalne vrednosti elemenata vektora promenljivih stanja usvoje jedinične vrednosti napona koji su pri tom u fazi.

Kako bi se rešio ovaj problem, za inicijalne vrednosti vektora $\boldsymbol{x}$ se mogu usvojiti vrednosti koje malo odstupaju od nominalnih:

$$\boldsymbol{x} = \begin{bmatrix} 0 & 5,70 & 8,60 & 1 & 0,95 & 1,05 \end{bmatrix}^T .$$

Za ovako usvojenu inicijalnu vrednost vektora stanja i usvojeni kriterijum konvergencije $\xi = 0{,}0001$, proračun konvergira u 6 iteracija. U tabeli 3 prikazane su vrednosti funkcije cilja (sume otežanih kvadrata reziduala merenja), kao i maksimalnog odstupanja vektora promenljivih stanja, kroz svih 6 iteracija.

Tabela 3 – Funkcija cilja i maksimalna odstupanja vektora $\boldsymbol{x}$ po iteracijama

Iteracija	$J(\boldsymbol{x})$	$\max\{\Delta\boldsymbol{x}\}$
1	624060,1	$1{,}25\cdot10^{-1}$
2	163442,3	$5{,}36\cdot10^{-2}$
3	7028,8	$2{,}01\cdot10^{-2}$
4	140,2	$3{,}87\cdot10^{-3}$
5	4,9	$1{,}55\cdot10^{-4}$
6	4,7	$2{,}48\cdot10^{-7}$

Proračunati vektor stanja ima sledeće vrednosti:

$$\boldsymbol{x} = \begin{bmatrix} 0 & -1,234 & 2,766 & 1,000 & 0,974 & 0,944 \end{bmatrix}^T .$$

Drugi problem korišćenja strujnih merenja u proračunu statičke estimacije stanja, uočava se ukoliko se za inicijalnu vrednost vektora $\boldsymbol{x}$ usvoji nešto drugačija vrednost:

$$\boldsymbol{x} = \begin{bmatrix} 0 & -5,70 & -8,60 & 1 & 0,95 & 1,05 \end{bmatrix}^T .$$

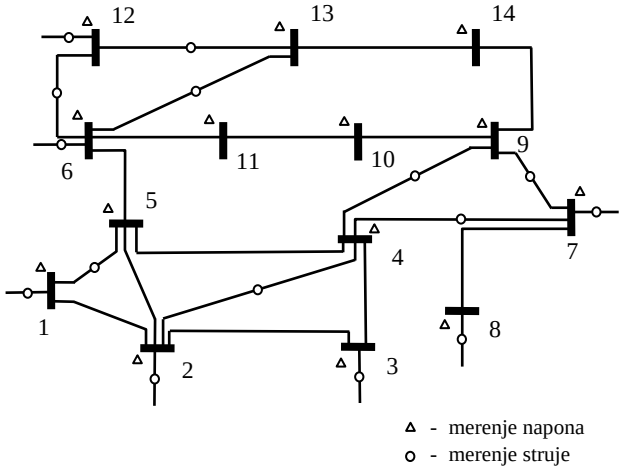
U ovoj situaciji proračun će ponovo konvergirati u 6 iteracija, sa sličnim vrednostima $J(\boldsymbol{x})$ i $\max\{\Delta\boldsymbol{x}\}$, ali ovaj put ka drugom rešenju:

$$\boldsymbol{x} = \begin{bmatrix} 0 & -1,234 & -2,766 & 1,000 & 0,974 & 0,944 \end{bmatrix}^T .$$

Ovim je potvrđeno da za dati skup raspoloživih merenja, postoje 2 rešenja statičkog estimatora stanja, odnosno dokazano je da u prisustvu strujnih merenja, sistem može biti nejedinstveno observabilan.

4.2 Test mreža sa 14 čvorova – sa strujnim merenjima

Test sistem sa 14 čvorova i raspored merenja, prikazani su na slici 2. Može se uočiti da u sistemu postoje merenja modula napona i struje potrošnje na svim sabirnicama, uz još 8 merenja modula struja u granama.



Slika 2 – Test mreža sa 14 čvorova

U tabeli 5 date su vrednosti korišćenih merenja modula struja po granama.

Tabela 5 – Vrednosti merenja modula struja po granama

Čvor 1	Čvor 2	I_{ij} [r.j.]
1	5	0,228
2	4	0,150
4	7	0,301
4	9	0,053
6	12	0,016
6	13	0,045
7	9	0,338
12	13	0,032

Izmereni moduli struje potrošnje imaju sledeće vrednosti: I_1 = 0,956, I_2 = 0,305, I_3 = 0,999, I_4 = 0,460, I_5 = 0,216, I_6 = 0,126, I_7 = 0, I_8 = 0,446, I_9 = 0,322, I_{10} = 0,104, I_{11} = 0,294, I_{12} = 0,061, I_{13} = 0,155, I_{14} = 0,154.

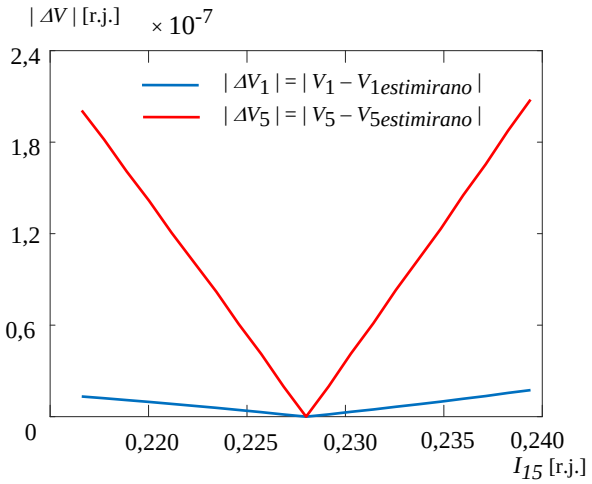
U ovom primeru, usled velikog broja i dobrog rasporeda merenja, proračun konvergira u samo 4 iteracije.

Iz ovoga se može zaključiti da postojanje merenja aktivnih i reaktivnih snage ne predstavlja neophodan preduslov za postojanje jedinstveno observabilnog sistema.

4.3 Analiza osetljivosti sistema sa 14 čvorova (bez merenja snage)

Na test sistemu iz prethodnog primera izvršena je analiza osetljivosti estimiranih napona u čvorovima 1 i 5, na promenu izmerene vrednosti struje u grani 1-5, slika 3.

Za relativnu grešku merenja biće usvojena vrednost koja iznosi ±5%, što znači da će se vrednost struje I_{15}, u ovom primeru kretati u opsegu od I_{15min} = 0,217 [r.j.], do I_{15max} = 0,239 [r.j.]. Proračun vrednosti napona čvorova 1 i 5, sproveden je za vrednosti merenja struje u grani 1-5 iz opsega od I_{15min} do I_{15max}, sa korakom od 0,00114 (20 tačaka).



Slika 3 – Analiza osetljivosti na test sistemu sa 14 čvorova (bez merenja snage)

Sa slike 3 se može primetiti da je osetljivost estimiranih vrednosti napona na grešku merenja modula struje zanemarljivo mala (reda veličine ~10^{-7}).

Situacija bi bila drastično drugačija kada bi se razmatrao uticaj greške merenja modula napona na estimirane vrednosti [4].

5. ZAKLJUČAK

U ovom radu izvršena su brojna zanemarenja, ali nije se izgubilo na opštosti samog proračuna. Osnovni zaključak vezan za statičku estimaciju stanja u prenosnim mrežama, jeste da je osnova za validne i upotrebljive rezultate, raspolaganje sa kvalitetnim skupom, dobro raspoređenih i redundantnih merenja, odnosno pravilno tretiranje raspoloživih merenja.

Objašnjen je tretman merenja različitog tipa, kao i metod minimuma sume otežanih kvadrata reziduala za rešavanje sistema nelinearnih, algebarskih i simultanih jednačina.

Ovakav pristup za rešavanje problema statičke estimacije stanja na realnim EES-ima ima i svoje nedostatke. Neki od tih nedostataka prikazani su kroz analizu uticaja strujnih merenja na proračun statičke estimacije. Zaključeno je da mnoga ranije ustanovljena pravila (kao što je činjenica da Jakobijan matrica punog ranga implicira jedinstveno rešenje statičkog estimatora stanja), ne važe u slučaju prisustva strujnih merenja.

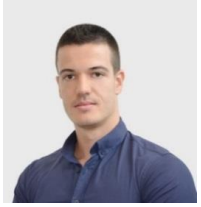
Velika dimenzionalnost sistema zahteva primenu tehnike retkih matrica za memorisanje matrice admitansi. Zbog toga je veoma bitno koristiti što efikasnije programske alate za faktorizaciju, množenje matrica i rešavanje sistema linearnih jednačina, sve to u cilju što bržeg i efikasnijeg proračuna koji predstavlja osnovu za sve druge energetske funkcije.

Takođe, u realnim EES-ima, postoje velike varijacije u dužinama vodova, što utiče na numeričku stabilnost proračuna, a svakako i na osetljivost estimiranih vrednosti napona čvorova na izmerene vrednosti u granama kojima su oni povezani. Upravo ova razmatranja predstavljala bi logičan nastavak ovog rada.

6. LITERATURA

1. A.T.Sarić, M.S.Ćalović: *Statička estimacija stanja u elektroenergetskim sistema*, Tehnički fakultet Čačak, Kraljevo, 2008.
2. A.Abur: *Power System State Estimation, Theory and Implementation*, Texas A&M University, College Station, Texas, U.S.A, 2004.
3. A.Monticelli: *State estimation in electric power systems, a generalized approach*, University of Campinas, Norwell, Massachusetts, U.S.A, 1999.
4. G.Švenda: *Specijalizovani softveri u elektroenergetici*, skripta sa predavanja iz istoimenog predmeta na master studijama, Fakultet Tehničkih Nauka, Novi Sad, 2016.

Kratka biografija:



Miroslav Pavlović rođen je u Novom Sadu 1993. godine. Diplomski rad na Fakultetu tehničkih nauka u Novom Sadu odbranio je 2016. godine u oblasti Elektrotehnika i računarstvo, smer Elektroenergetski sistemi. Iste godine upisuje master studije na istom smeru.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/07BE04Adamovic

AUTOMATIZACIJA TESTIRANJA PRORAČUNA KRATKIH SPOJEVA

AUTOMATION OF SHORT CIRCUIT CALCULATION TESTING

Milovan Adamović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je izvršena automatizacija testova koji vrše proračun kratkih spojeva koristeći softversku funkcionalnost za proračun. Najpre su objašnjeni osnovni principi testiranja softvera, osnovni principi automatskog testiranja i na kraju sama realizacija automatskih testova. Predstavljen je algoritam koji se koristi za postavku šema u željeno stanje pre samog proračuna kratkih spojeva kao i rezultati izvršenih testova.*

Ključne reči: *kratki spojevi, testiranje softvera, automatsko testiranje.*

Abstract *–In this paper, the automation of tests that calculate short circuit using software function for calculating. First, the basic principles of software testing, the basic principles of automatic testing, and finally the realization of automatic tests, are explained. The algorithm used to set schematics to the desired state before calculate short circuit is presented, as well as the results of the performed tests.*

Keywords: *short circuit, software testing, automatic testing.*

1. UVOD

Testiranje softvera je proces koji se koristi da bi se utvrdila ispravnost, potpunost i kvalitet razvijenog softvera. Nepravilno kreiran softver može imati velike posledice. Ukoliko je sam softver namenjen za komercijalnu upotrebu i ukoliko se ne ponaša očekivano može izazvati novčane gubitke i silne penale od strane korisnika.

Testiranje softvera je način da se obezbedi manji broj grešaka, manji trošak održavanja i sveukupne cene softvera. Testiranje je aktivnost koja se sprovodi radi evaluacije kvaliteta proizvodnje softvera i njegovog poboljšanja.

Ono nije aktivnost koja počinje samo nakon kompletiranja faze kodiranja. Softversko testiranje se danas vidi kao aktivnost koja obuhvata ceo proces razvoja i održavanja, i predstavlja važan deo kompletne konstrukcije softvera. Planiranje testiranja treba da počne sa ranom fazom izrade zahteva za kvalitet softverskog proizvoda i procesa njegovog projektovanja, i test planovi i procedure moraju biti sistematski i kontinualno razvijani i po potrebi redefinisani.

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

Pravi stav prema kvalitetu je prevencija, mnogo je bolje izbeći probleme nego ih ispravljati, kao i za sve ostale životne situacije fraza je neizbežna da je bolje sprečiti, nego lečiti [2].

U radu je predstavljena problematika i pronalaženja najoptimalnijeg rešenja za pripremu šema od interesa za proračun kratkih spojeva pomoću automatskih testova. Algoritam rešenja prikazan je na blok dijagramu i opisno je objašnjen princip realizacije.

Rad je organizovan tako da su u drugom delu objašnjeni modeli procesa razvoja softvera, dok su u trećem delu objašnjeni nivoi testiranja i test dizajn tehnike. U četvrtom delu objašnjen je postupak za proračun kratkih spojeva. U petom delu objašnjen je način realizacije automatskih testova za potrebe rada i algoritma primenjenog u testovima koji vrše postavku šema sistema. Na kraju su dati predlozi za proširenje rada kroz opis u zaključku.

2. MODELI PROCESA RAZVOJA SOFTVERA

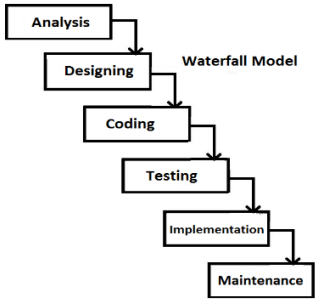
Modeli procesa razvoja softvera su različiti procesi i metodologije, koji se biraju i koriste u razvoju projekta, u zavisnosti od ciljeva i zahteva projekta. Postoji više modela koji su razvijeni tokom godina, svaki sa drugačijim ciljevima. Uopšteno gledano modeli specificiraju različite faze u procesu razvoja, kao i redosled kojim se te faze izvršavaju. Odabir modela koji se koristi u razvoju softvera ima ogroman uticaj na testiranje koje će se sprovesti. Odabrani model definiše šta, kada i kako će se testirati i utiče na odabir tehnika koje će se koristiti. Navedeni su i opisani samo neki od postojećih modela [2].

2.1. Waterfall model (model vodopada)

Waterfall model (model vodopada) je prvi model koji je uveden u razvoj softvera. Često se naziva još linearni sekvencijalni model. Veoma je lak za razumevanje i upotrebu. Prema ovom modelu svaka faza mora u potpunosti biti kompletirana pre nego što počne sledeća, odnosno faze se ne mogu preklapati [2].

Na slici 2.1 prikazan je blok dijagram waterfall modela razvoja softvera.

Nakon završetka jedne faze započinje druga, sa testiranjem se kreće nakon sto je razvoj kompletno završen. Prednosti modela su da je lak za razumevanje i upotrebu, nema preklapanja faza, pogodan je za manje projekte gde su zahtevi jasni. Mane modela su te da se operativni softver dobija kasno, veliki rizik i neizvesnost, ukoliko je aplikacija u fazi testiranja teško se vratiti nazad i nešto promeniti u fazi razvoja, loš u slučaju rizika od čestih promena zahteva.

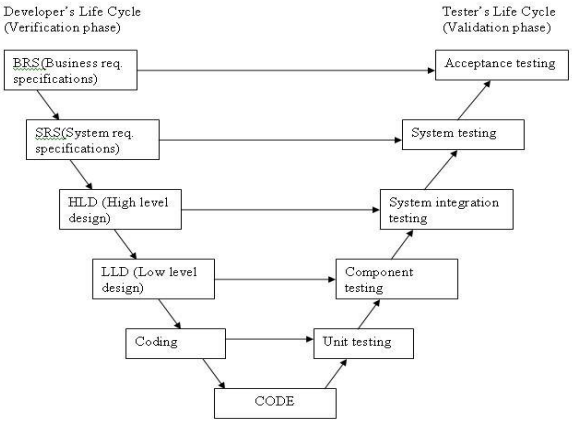


Slika 2.1 *Waterfall model razvoja softvera*

Navedeni model se koristi kada su zahtevi poznati jasni, kada je tehnologija u kojoj se razvija poznata i kada je projekat kratak. U ovakvom načinu razvoja nema interakcije sa klijentima tj. krajnjim korisnicima, tek kada je projekat potpuno gotov može se pokazati korisnicima. U slučaju da tada dođe do ozbiljnih grešaka cena ispravke je obično velika [2].

2.2. V model

Ovaj model je poznat i pod nazivom verifikacija i validacija. Svaka faza mora biti kompletirana pre nego sto počne sledeća faza u razvoju modela. Testiranje se planira paralelno sa razvojem.
Sam model se prikazuje u obliku karakterističnog slova V. Zahtevi se analiziraju na početku modela. Pre nego što se krene sa razvojem kreira se plan sistemskog testiranja. Ovaj plan testiranja se fokusira na specificirane funkcionalnosti koje su definisane u fazi prikupljanja zahteva. Faza dizajna visokog nivoa se fokusira na arhitekturu i dizajn sistema. Pruža pregled rešenja platformi sistema, servisa koje je potrebno implementirati. Paralelno se kreira plan integracionog testiranja kako delovi softvera rade zajedno. Faza dizajna niskog nivoa se fokusira na dizajn softverskih komponenti. Definiše logiku za svaku komponentu sistema, i kreira se klasni dijagram sa svim metodama i relacijama između klasa. Paralelno se kreira plan testiranja komponenti [2].
Na slici 2.2 prikazan je V model razvoja softvera.



Slika 2.2 *V model razvoja softvera*

Faza implementacije je faza gde se odigrava kompletno pisanje koda. Kada je kodiranje završeno putanja izvršavanja kreće desnom stranom V modela, gde se kreirani test planovi stavljaju u upotrebu. Kodiranje se nalazi na dnu V modela, gde se dizajn komponenti pretvara u kod. Jedinično testiranje izvršavaju programeri koji rade na kodu paralelno sa pisanjem koda. Prednosti modela su da je jednostavan i lak za upotrebu, određene aktivnosti testiranja se dešavaju pre kodiranja na primer planiranje i dizajn testova sto štedi dosta vremena, defekti se mogu pronaći rano, dobar za projekte sa jasnim zahtevima. Mane modela su te da je veoma rigidan i nefleksibilan, softver se razvija unutar faze implementacije pa ne postoje rani prototipovi koji se mogu pokazati klijentu, ukoliko se desi promena na pola puta u modelu svi test dokumenti se moraju ažurirati zajedno sa zahtevima.

3. AUTOMATSKO TESTIRANJE

U ovom poglavlju su predstavljeni nivoi testiranja i test dizajn tehnike, objašnjene su samo neke od navedenih.

3.1. Nivoi testiranja

Komponentno testiranje može da se primeni i izolovano od ostatka softvera u zavisnosti od faze procesa razvoja softvera. Zbog nedostatka ostalih komponenti softvera, vrše se različite simulacije razmene podataka [4].

Predmet testiranja mogu da budu funkcionalne karakteristike komponente, ali i nefunkcionalne karakteristike kao što je: ponašanje hardvera (curenje memorije), performanse.
Integraciono testiranje testira prenos informacija između različitih komponenti softvera, interakciju između različitih delova sistema kao što su: operativni sistem, baza podataka i hardver. Integraciono testiranje treba da bude sprovedeno od strane integracionog inženjera ili test inženjera integracije [4].
Sistemsko testiranje uzima u obzir ponašanje krajnjeg produkta/softvera u skladu sa definisanim ciljem projekta koji je bio predmet razvoja. Sistemsko testiranje može da obuhvata testove vezane sa specifične zahteve koje je softver trebao da ispuni, biznis proces, specifično korišćenje softvera, interakciju sa operativnim sistemom ili bazom podataka. Sistemsko testiranje je najčešće završna faza u procesu testiranja koje treba da potvrdi da softver ispunjava sve zahteve pre nego što ser isporuči klijentu, svrha sistemskog testiranja takođe može da bude i pronalaženje što je moguće više grešaka.
Prihvatljivo testiranje treba da odgovori na nekoliko pitanja: „Da li softver može biti isporučen?“, „Šta su, i da li ima rizika?“, „Da li je razvojni tim ispunio sve svoje obaveze?“. Prihvatljivo testiranje najčešće izvršavaju sami klijenti ili korisnici, neretko u saradnji sa test inženjerima. Prihvatljivo testiranje, isto kao i sistemsko testiranje, zahteva okruženje za testiranje koje preslikava okruženje u kojem će softver raditi u produkciji. Glavni cilj prihvatljivog testiranja je da pruži sigurnost i pouzdanost u softver, deo softvera ili neke nefunkcionalne karakteristike kao što su performanse [1].

3.2. Test dizajn tehnike

Statičke tehnike za dizajn testova – Većina statičkih tehnika se koristi prilikom testiranja bilo kakve dokumentacije vezane za softver ili izvorni kod softvera. Predmet testiranja statičkih tehnika mogu biti dizajn dokumenti i dizajn model softvera, funkcionalne

specifikacije i specifikacije zahteva koje softver treba da ispuni.

Tehnike „crne kutije“ – Prva dinamička tehnika koja će biti predstavljena je tehnika „crne kutije“ ili tehnika „unos/rezultat“. Naziv „crna kutija“ potiče od samog načina pristupa softveru, softveru se pristupa kao crnoj kutiji koji ima ulaz i izlaz(rezultat) i nije poznato kako su komponente softvera struktuirane unutar softvera, takođe nije poznato kako softver izračunava krajnji rezultat. Prilikom testiranja softvera tehnikom „crne kutije“, test inženjer se fokusira na čemu softver zaista radi, a ne na koji način softver radi [1].

Tehnike „bele kutije“ – Tehnike „bele kutije“ koriste unutrašnju strukturu softvera za izvršavanje testova. Naziv „bela kutija“ ili „staklena kutija“ potiče od zahteva za poznavanjem načina implementacije softvera i načinom rada softvera. Na primer, tehnika „bele kutije“ testira izvršavanje petlji unutar izvornog koda softvera. Različiti testovi mogu biti dizajnirani da pojedine petlje, unutar softvera izvornog koda, budu izvršene jednom, dva puta ili više puta zaredom. Testiranje tehnikom „bele kutije“ može biti izvršeno bez obzira na funkcionalnost softvera [1].

Tehnike koje se zasnivaju na iskustvu test inženjera – Iskustvo, znanje, veštine test inženjera najviše doprinose testiranju ovom tehnikom. Tehničko i poslovno iskustvo test inženjera je veoma bitno, jer donose različite perspektive u analizi testova i procesu dizajna. Na osnovu prethodnog iskustva sa sličnim softverima, test inženjer može da predvidi kritičan deo softvera koji je veoma bitan za testiranje.

4. Funkcionalnost proračuna kratkih spojeva

Funkcija za proračun struja kratkih spojeva je jedna od jako bitnih funkcija, na osnovu čijih rezultata se bira relejna oprema setuju vrednosti struje zaštite i mnoge druge operacije u mreži.

4.1 Algoritam za proračun kratkih spojeva

Sama funkcija za proračun struja kratkog spoja realizovana je pomoću algoritma koji se sastoji od sledećih koraka:

Svaki proračun kratkog spoja počinje sa dekompozicijom mreže pod kvarom na deo pre kvara i fiktivno takozvano delta kolo. Ovo je primer tj metoda superpozicije. Fiktivno delta kolo je iste topologije kao i originalno kolo osim što je pasivno u celini osim na mestu kratkog spoja gde postoje strujni i naponski izvori, u ostatku kola su izvori pasivni s tim da naponski izvori predstavljaju kratak spoj a strujni izvori predstavljaju prekid [3].

Određivanje matrice ekvivalentnih Tevenenovih impedansi viđenih sa svakog mesta kvara kao i uzajamnih ekvivalentnih Tevenenovih impedansi između mesta kvara. Potrebno je računati ove impedanse za sve tri faze ukoliko se radi sa trofaznim balansiranim mrežama ili se računaju samo za jednu fazu ukoliko mreža nije balansirana. Zbog ovoga se svi elementi na mreži (transformatori, generatori, prekidači i ostali) zamenjuju sa ekvivalentnim impedansama. Impedansa transformatora sa regulacionom sklopkom se računa uzimajući u obzir stvarni položaj regulacione sklopke [3].

Definišu se opcije za selektovani tip kratkog spoja, pri tome se setuje period proračuna, minimalna i maksimalna struja kratkog spoja, snaga mreže, otpor impedanse na mestu kvara, tip kvara [3].

Računanje struje kratkog spoja i kompenzacija struje koristeći kanonički model [3].

Računanje stanja delta kola. Za ovu svrhu se koristi modifikovani Shirmohammadi -ev algoritam za proračun struje. U ovom algoritmu svi elementi mreže su zamenjeni sa svojim impedansama. Potrošači su izostavljeni i mesto kvara je modelovano kao idealni strujni izvor, i predstavlja jedini generator u kolu. Posmatrano kolo je tada linearno i može se rešiti direktno bez primene iteracija. Takvim postupkom je potrebna samo jedna iteracija tj. proračun kako bi se rešilo delta kolo [3].

Nakon proračuna za delta kolo, celokupno stanje mreže u kvaru se dobija superpozicijom kola pre kratkog spoja i fiktivnog delta kola [3].

5. AUTOMATIZACIJA PRORAČUNA KRATKIH SPOJEVA

Razvoj testova za potrebe automatizacije realizovan je u okviru okruženja PyCharm. Koristi se za programiranje u pajtonu. Razvojno okruženje pruža analizu koda, grafički debager, integrisanu jedinicu tastera i podržava veb razvoj. Dobra osobina okruženja jeste ta da podržava rad sa json fajlovima. Samo razvojno okruženje ima instalirane pakete koji mu omogućavaju rad sa json fajlovima. Json je tip formata fajla koji se koristi za čuvanje i razmenu podataka. Jednostavan je za rad jer je korišćenje veoma jednostavno, lak je upis i čitanje. U okviru same realizacije rada json fajl pored standradnih instalacionih putanja, za potrebe rada sadrži i ključeve koji su od interesa. Razlog zbog čega se koristi u realizaciji rada je taj da ukoliko se desi neka promena u konfiguraciji, ključevi elemenata se promene izmena se veoma lako ostvaruje menjajući ključ u json-u umesto izmene automatskih testova, ostvaruje se ušteda na vremenu potrebnom za održavanje testova.

Na slici 5.1 prikazana je struktura json fajla.


```
{
    "Rail Booking": {
        "reservation": {
            "ref_no": 1234567,
            "time_stamp": "2016-06-24T14:26:59.125",
            "confirmed": true
        },
        "train": {
            "date": "07/04/2016",
            "time": "09:30",
            "from": "New York",
            "to": "Chicago",
            "seat": "57B"
        },
        "passenger": {
            "name": "John Smith"
        },
        "price": 1234.25,
        "comments": ["Lunch & dinner incl.", "\"Have a nice day!\""]
    }
}
```


Slika 5.1 *Prikaz strukture json fajla*

Kako bi problematika automatizacije proračuna kratkih spojeva bila rešena, sam koncept realizacije osmišljen je tako da su automatski testovi podeljeni u dve celine: testovi zaduženi za postavku seme za stanice od interesa u kojima se simuliraju kratki spojevi, test koji je zadužen za proračun kratkih spojeva nad već pripremljenim šemama i upis vrednosti u željeni fajl.

U okviru testova zaduženih za postavku šema neophodno je ispuniti par zahteva: postaviti sve regulacione transformatore u neutralni položaj, razvezati sve potrošače i kondezatore sa sabirnica. Da bi realizovali postavku sistema potrebno je sve otvorene elemente na postojećoj šemi vratiti u zatvoreni položaj.

To se čini zbog toga što su na taj način zatvoreni strujni krugovi od izvora napajanja do potrošača i jednostavnim manipulacijama, mogu se raskačiti svi neželjeni elementi. Kako bi izbegli pristupanju svim elementima jer je cilj imati sve elemente u stanju zatvoren na datoj stanici iskorišćena je pomoć trace funkcionalnosti koja se nalazi u sklopu samog softvera.

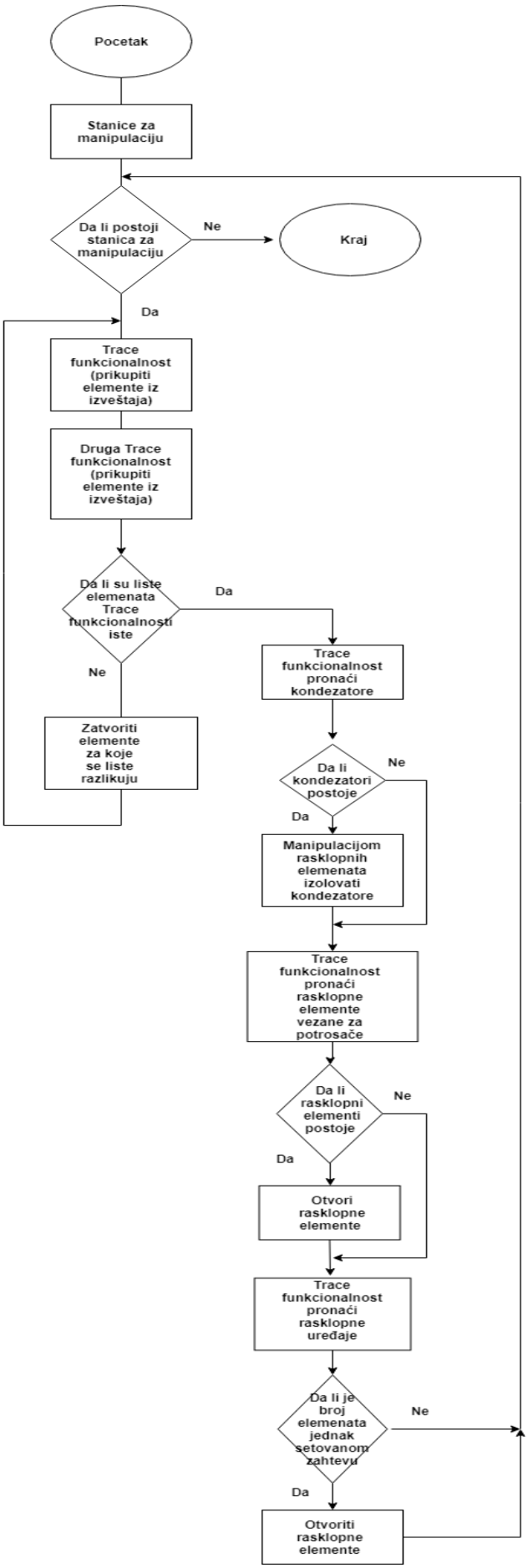
Navedena problematika rešena je korišćenjem dve trace funkcionalnosti. Trace funkcionalnosti kao startnu lokaciju koriste izvore napajanja i prikupljaju sve elemente nadole od izvora napajanja, za izabrani smer funkcionalnosti nadole.

Korišćenjem funkcionalnosti i upoređivanjem elemenata prikupljenih ih izveštaja funkcionalnosti nakon filtriranja sadržaja izveštaja kako bi bilo izbegnuto manipulisanje sa nepotrebnim elementima prikupljeni su elementi od interesa.

Razlika sadržaja dve funkcionalnosti predstavlja listu elemenata koji se nalaze u stanju otvoren i nad njima je neophodno manipulisati. Samo stanje tih elemenata neophodno je promeniti. Ukoliko je zateknuto stanje takvo da su svi elementi u stanju zatvoren pri prvom prolasku algoritma ne postoji potreba za manipulacijom nad elementima, ukoliko postoje elementi za manipulaciju menja im se stanje i ponovo se ponavlja isti postupak sa pokretanjem dve trace funkcionalnosti ako tada ne postoji razlika elemenata postupak se prekida ako naprotiv i dalje postoje elemenati za manipulaciju njima će se manipulisati i tako dok se svi elementi ne postave u stanje zatvoren.

Nakon ovakve postavke željenog stanja primenom trace funkcionalnosti pronalaze se svi kondezatore koji postoje i razvezuju se od ostatka šeme, isto se izvršava i za potrošače koji se odvajaju od ostatka šeme. Ovakvim pristupom pripremljena je šema od interesa za proračun kratkog spoja. Na slici 5.2 prikazan je blok dijagram algoritma za pripremu šema od interesa za proračun kratkih spojeva.

Test kojim se realizuje proračun kratkih spojeva koristi postavljeno stanja sistema ostvareno testom za postavku stanja. Stanje sistema se učitava i pronalaze se lokacije na kojima se simulira proračun kratkog spoja. Te lokacije se setuju u opcijama funkcionalnosti kratkog spoja, podešavaju se parametri funkcionalnosti koji su prikupljeni iz unapred definisanog csv fajla, računaju se vrednost struja kratkog spoja, date vrednosti se prikupljaju iz izveštaja funkcije. Sve prikupljene vrednosti se upisuju u dati csv fajl na tačno definisanu lokaciju, za dati tip kratkog spoja, željenu stanicu i postavku sistema. Isti proces se ponavlja za svaku stanicu ponaosob. Na slici 5.3 prikazana je topološka struktura šema stanica od interesa.

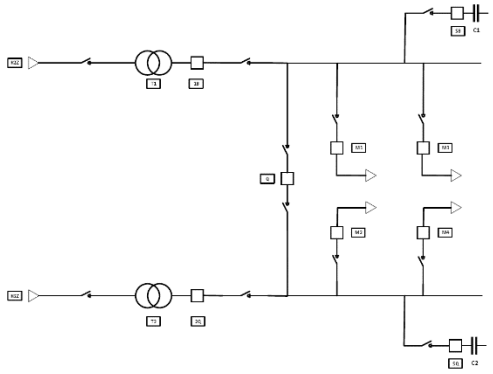


Slika 5.2 *Blok dijagram algoritma za postavku šeme za stanice od interesa*

5.1. Prednosti i mane automatskog testiranja vezane za opisani rad

Prednosti ostvarene automatizacijom proračuna: izbegnuto manuelno izvršavanje testova za proračun kratkih spojeva, izbegnuto popunjavanje elemenata kroz json, auto-

matizovana postavka šeme i priprema sistema za proračun bez obzira na zatečeno stanje sistema, jednostavnom manipulacijom se broj stanica za proračun može proširiti, izbegnute izmene na šemama korišćenjem metoda za generisanje pomeraja, testiranje performansi odziva softvera.



Slika 5.3 *Topološka struktura šema stanica od interesa*

Mane ispoljene automatizacijom testova: testovi za pripremu stanja sa porastom broja stanica se vremenski produžavaju, mogu trajati i nekoliko desetina sati u zavisnosti od broja stanica za pripremu, usporavanje testnog sistema.
Glavna svrha napisanih testova jeste održavanje. Zbog velikog broja manipulacija nad stanicama od interesa vreme izvršavanja testova vremenski će oscilirati.

6. ZAKLJUČAK

Proračun kratkih spojeva realizovan je podelom automatskih testova na dve grupe. Prva grupa je skup od dva testa koji priprema šemu na stanicama od interesa u okviru posmatranog sistema. Drugu grupu predstavlja test koji radi proračun struja kratkih spojeva za stanice od interesa koristeći sačuvane postavke sistema kreiranih testovima zaduženim za pripremu šema za proračun. Test vrši i upis proračunatih vrednosti u željeni fajl. Postoje dva dodatna testa koji rade verifikaciju fajla, ukoliko neki podaci nisu adekvatno popunjeni ili su popunjeni pogrešno, test skreće pažnju kako bi ispravili pogrešno, kao i test za proveru upisanih proračunatih struja spram definisanog kriterijuma.

Ukoliko vrednosti odstupaju podaci koji su nevalidni biće obeleženi. Vremenska zavisnost testova za pripremu šema veća je ukoliko se proširi broj stanica za manipulaciju. Mogući nastavci rada su primena postojećih testova za postavku šema sa manjim ili većim izmenama kako bi se automatizovale i ostale funkcionalnosti vezane za proračune, optimizacija datog rešenja ili pronalaženje boljeg kako bi se izvršavanje postavke šema ubrzalo i samim tim i testovi.

7. LITERATURA

[1] R.Black, Erik Van Veenendaal, D.Graham: *Foundations of Software Testing 3rd Edition*, South – Western, 2009.

[2] Miodrag Živković: Testiranje Softvera, Beograd 2018.

[3] Dragan Popović, Duško Bekut, Valentina Treskanica: Specijalizovani DMS Algoritmi, Novi Sad 2004.

[4] Miloš Kajkut: Automatsko Testiranje CVC DMS funkcije implementirane u aDms softver, Novi Sad 2018.

Kratka biografija:

Milovan Adamović rođen je u Šapcu 1993. god. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva - Energetska elektronika i Električne mašine odbranio je 2017. god. Iste godine upisao se na master studije na istom smeru.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE05Ilic

APLIKACIJA ZA UPRAVLJANJE RAZMENOM STUDENATA PUTEM ERASMUS+ PROGRAMA

STUDENT EXCHANGE APPLICATION THROUGH ERASMUS+ PROGRAMME

Jelena Ilić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – SOFTVERSKO INŽENJERSTVO I INFORMACIONE TEHNOLOGIJE

Kratak sadržaj – *U radu je objašnjen proces prijavljivanja studenata za program mobilnosti. Dat je model sistema kao i objašnjenje dela njegove implementacije. Prikazan je korisnički interfejs sistema.*

Ključne reči: *Erasmus+ program, razmena studenata, RESTful veb servisi, pozadinsko izvršavanje zadataka*

Abstract – *This paper explains the process of student application for a mobility program. The implementation and model of the system are given. User interface of the system is shown.*

Keywords: *Erasmus+ programme, student exchange, RESTful web services, background task execution*

1. UVOD

Erasmus+ [1] program je program Evropske unije koji podržava obrazovanje, obuku i mobilnost mladih širom Evrope. Povećava mogućnosti zapošljavanja kao i modernizacije obrazovnih sistema. Ovim putem se otvara mogućnost putovanja u edukativne svrhe za učenike, studente, nastavno osoblje i omladinske radnike.

Erasmus+ program nudi finansijsku podršku organizacijama i institucijama koje rade u oblasti obrazovanja. Program traje 7 godina (od 2014. do 2020. godine) i njegov budžet od 14,7 milijardi evra pruža mogućnosti za preko 4 miliona Evropljana da studiraju, obučavaju se i stiču nova iskustva u inostranstvu.

U radu će biti posmatrana procedura podnošenja zahteva studenta na Fakultetu tehničkih nauka u Novom Sadu za studiranje u inostranstvu. Cilj rada jeste automatizacija celog procesa kako bi se studentu olakšalo prijavljivanje na Erasmus+ program, tako i organizaciji fakulteta odobravanje i uvid u sve zahteve studenata koji žele da dobiju priliku za studiranje u inostranstvu. Takođe u radu će biti opisane korišćene tehnologije za implementaciju rešenja ali i objašnjena sama implementacija određenih delova sistema.

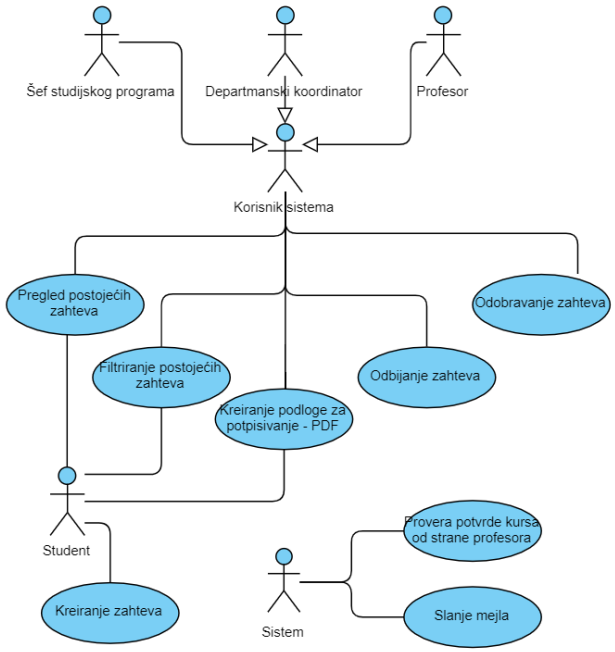
2. SPECIFIKACIJA ZAHTEVA SISTEMA

Ova aplikacija treba da podrži funkcionalnosti vezane za čitav proces koji student treba da prođe prilikom konkurisanja na program razmene.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Goran Savić.

2.1. Slučajevi korišćenja

Pomoću *dijagrama slučajeva korišćenja* moguće je prikazati skup akcija (slučajeva korišćenja) koje korisnici (akteri) sistema mogu da izvrše (slika 2.1).



Slika 2.1 *Dijagram slučajeva korišćenja*

2.2. Uloga sistema u okviru aplikacije

S obzirom da sistem ima posebnu ulogu u aplikaciji, funkcionalnosti koje može da obavi su slanje e-mail poruka i izvršavanje određenih pozadinskih zadataka.

2.2.1 E-mail podrška

Po odobravanju predmeta od strane koordinatora sistem automatski šalje e-mail poruke svim profesorima čije je predmete student naveo u svojoj prijavi.

2.2.1 Izvršavanje zadataka u pozadini

Automatska saglasnost profesora nakon određenog vremenskog perioda obezbeđena je uz pomoć pozadinskih zadataka. To znači da se vrši provera potrvrde profesora da se njegov predmet zameni na stranom fakultetu.

3. TEHNOLOGIJE

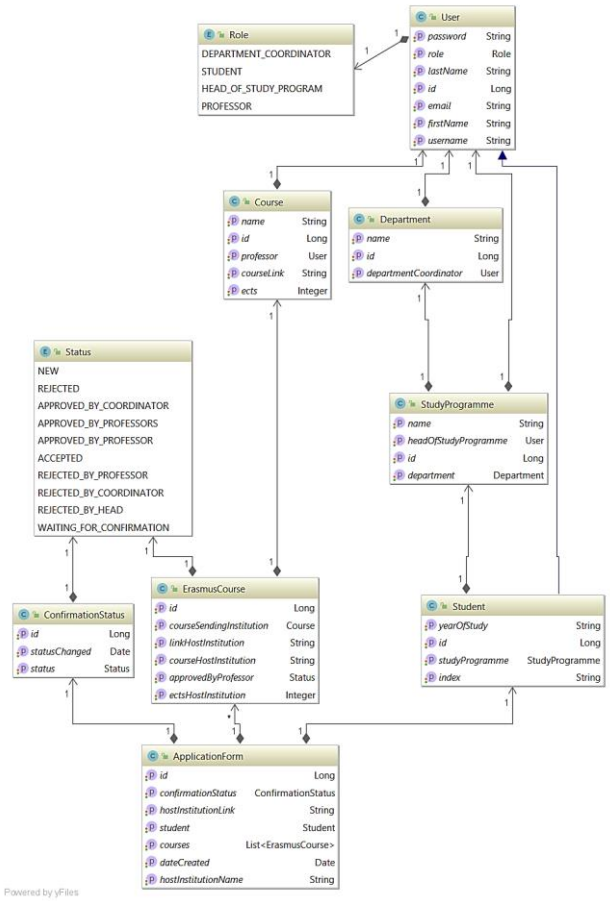
Aplikacija je napisana u programskom jeziku *Java* [2]. Za implementaciju serverske strane aplikacije korišćen je *Spring Boot*, koji olakšava korišćenje *Spring* [3] radnog okvira, a izgrađena je pomoću alata *Apache Maven* [4]. Na klijentskoj strani korišćen je *Angular* [5]. Skladištenje podataka omogućeno je putem *MySQL* [6] baze podataka.

4. MODEL SISTEMA

Kada se govori o modelu čitavog sistema potrebno je sagledati sve aspekte. Počinje se od modela podataka. S druge strane aplikacija prolazi kroz različite procedure i potrebno je obratiti pažnju na model poslovnog procesa.

4.1 Model podataka

Model podataka koji se preslikava na prethodno pomenutu bazu podataka prikazan je na slici 4.1 u okviru dijagrama klasa. U priloženom dijagramu prikazana su kako svojstva svakog entiteta tako i relacije među njima.

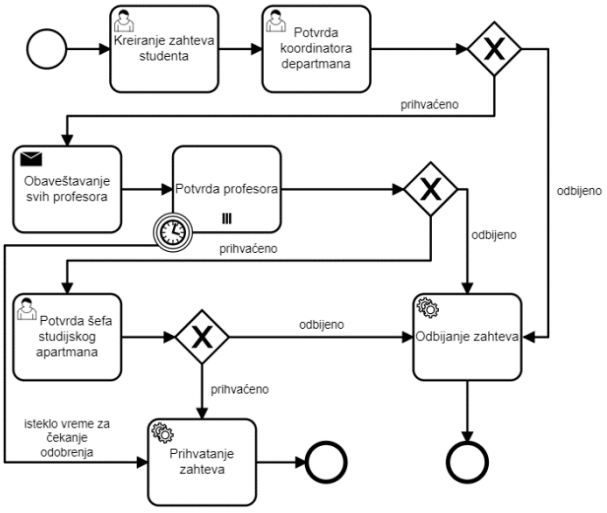


Slika 4.1 *Dijagram klasa osnovnog modela*

Kreiranje novog zahteva studenta opisano je klasom *ApplicationForm*. S obzirom da student prilikom prijavljivanja na program razmene treba da odabere predmete, ovaj entitet ima vezu *1:N* sa entitetom *ErasmusCourse*. U okviru njega definisani su svi potrebni podaci koje student treba da popuni za jedan predmet. Entitet *Course* opisuje postojeće predmete na Fakultetu tehničkih nauka sa podacima o broju bodova, profesoru i linku gde je opis predmeta dostupan. Uloge korisnika sistema modelovane su enumeracijom *Role* sa vrednostima *Student, Professor, Department-Coordinator, Head-Of-Study-Program.*

4.2 Model poslovnog procesa

Model poslovnog procesa predstavlja bitan deo sistema u kome se smenjuju unapred predviđeni događaji, odnosno aktivnosti. Ovakav model prikazuje redosled događaja tokom obavljanja funkcionalnosti aplikacije, odnosno sekvencijalne i konkurente korake u jednom poslovnom procesu. (slika 4.1).



Slika 4.1 *Model poslovnog procesa*

5. IMPLEMENTACIJA

Kada se govori o funkcionisanju veb aplikacija potrebno je pomenuti osnovne koncepte koji omogućavaju komunikaciju. Drugi deo se odnosi na implementaciju određenih delova aplikacije.

5.1 RESTful web servisi

Serverska strana aplikacije implementirana je pomoću Spring radnog okvira, a zasniva se na REST (Representational State Transfer) protokolu [7]. REST definiše skup principa za razvoj veb usluga. Implementacija RESTful web servisa predstavljena je sa 4 osnovna principa: eksplicitno korišćenje HTTP metoda, stateless komunikacija, identifikacija resursa pomoću URI mehanizma i transfer podataka putem XML ili JSON objekata.

5.2 Implementacija serverskog dela aplikacija

Pre svega, potrebno je opisati tačku sistema u kojoj se vrši komunikacija sa bazom podataka. Pored toga, svaka aplikacija zahteva određenu obradu podataka. Implementacija funkcionalnosti predstavlja poslovnu logiku aplikacije. Zadatak serverskog dela aplikacije je i da omogući dostupnost javnog API-ja preko koga se omogućava komunikacija klijenata sa serverom.

5.3 Implementacija klijentskog dela aplikacija

Klijentska aplikacija se sastoji od komponenti TypeScript klasa predstavlja aplikativnu logiku. Ima pridružen *template* fajl (html) koji definiše izgled komponente. Klasa komunicira sa prikazom kroz javni API odnosno preko definisanih atributa i metoda. Servis klijentske aplikaciju predstavlja tačku gde se vrši komunikacija sa serverskim delom aplikacije.

5.4 Autentifikacija pomoću JWT tokena

JWT (JSON Web Token) [9] predstavlja način autentifikacije korisnika u sistemu. Princip funkcionisanja ovakve autentifikacije je da svaki korisnik po uspešnoj prijavi na sistem dobija jedinstveno izgenerisan token od strane servera. Taj token korisnik dalje koristi u radu sa aplikacijom, tako što ga prosleđuje u svakom sledećem zahtevu. Time je korisniku omogućen pristup rutama i resursima. U okviru klijentske aplikacije omogućeno je presretanje zahteva i dodavanje tokena u svaki zahtev ka serveru. Na serverskoj strani se vrši provera postojanja tokena pri svakom zahtevu kao i njegove validnosti.

5.5 Servis za slanje e-mail poruka

Slanje e-mail poruka često predstavlja blokirajuću tačku u sistemu. S obzirom da student može da odabere proizvoljan broj predmeta, dolazi se do problema slanja velikog broja poruka u istom trenutku. Čekanje na završetak ove operacije nije dobro rešenje. Iz tog razloga korišćeni su asinhroni zadaci.

Anotacija *@Async* iznad metode za slanje poruka označava da će se metoda izvršavati u posebnoj niti. Asinhrono izvršavanje operacija potrebno je omogućiti i pomoću anotacije *@EnableAsync* u main klasi aplikacije.

```
@Async("threadPoolTaskExecutor")
public void sendConfirmationEmail(String to, String
subject, String text) {
 try {
  SimpleMailMessage message = new
SimpleMailMessage();
   message.setTo(to);
  message.setSubject(subject);
  message.setText(text);
  emailSender.send(message);
 } catch (MailException exception) {
  throw new
BadRequestException(exception.getMessage());
 }
}
```

Listing 5.3 *Implementacija slanja e-mail poruka*

5.6 Čuvanje istorije statusa zahteva studenata

Istorija promene statusa predstavlja kolekciju svake promene statusa zahteva. Za svaku promenu čuva se status u koji je promenjen kao i trenutno vreme promene.

5.7 Implementacija pozadinskih zadataka

Trenutak potvrde koordinatora beleži se u bazi podataka u okviru istorije promene statusa. Potrebno je obezbediti mehanizam provere koliko je vremena prošlo od tog trenutka i obezbediti automatsku potvrdu od strane profeora nakon isteka određenog vremena.

Zakazivanje zadataka implementirano je uz pomoć *@Scheduled* anotacije (listing 5.4).

```
@Scheduled(cron = "0 * * ? * *")
@Transactional
public void checkProfessorsApproval(){
 List<ApplicationForm> applicationForms =
applicationFormRepository.
findByConfirmationStatusStatus(Status.
APPROVED_BY_COORDINATOR);

 for (ApplicationForm applicationForm : a) {
 long diff = new Date().getTime() -
applicationForm.getConfirmationStatus()
           .getStatusChanged().getTime();
 int diffDays = (int) (diff / DAY_IN_MS);
 if (diffDays >= WAITING_PERIOD) {
  ConfirmationStatus confirmationStatus =
confirmationStatusRepository
.getOne(a.getConfirmationStatus().getId());
confirmatonStatus.setStatusChanged(newDate())
confirmationStatus.setStatus(
Status.APPROVED_BY_PROFESSORS);
a.setConfirmationStatus(confirmationStatus);
confirmationStatusRepository.save(confirmationStatus
);
 }
}
```

Listing 5.4 *Implementacija pozadinskog zadatka*

Takođe, da bi se omogućilo izvršavanje zadataka potrebno je dodati anotaciju *@EnableScheduling* u okviru main klase aplikacije. Ovom konfiguracijom, pri pokretanju aplikacije startuju se sve metode koje su ovako anotirane.

Za definiciju vremenskog perioda kada će se vršiti provera korišćen je *cron* izraz. Format ovakvih izraza je: *sekunde, minuti, sati, dan u mesecu, mesec, dan u nedelji, godina*. U primeru je definisano da se zadatak, odnosno provera izvršava svakog dana u ponoć.

5.8 Generisanje podloge za potpisivanje

Podloga za potpisivanje predstavlja dokument koji služi za priznavanje perioda mobilnosti studenta. Svi odobreni predmeti se nalaze u ovom dokumentu, a sam dokument mora biti potpisan od strane koordinatora departmana i šefa studijskog programa. Svi korisnici sistema mogu da preuzmu ovu podlogu u PDF formatu.

S obzirom da se radi sa promenljivim skupom podataka potreban je mehanizam za dinamičko generisanje dokumenta u trenutku zahteva korisnika. Na serverskoj strani korišćena je *itext* biblioteka. Preuzimanje dokumenta na klijentskoj strani obezbeđeno pomoću biblioteke *filesaver* koja omogućava preuzimanje dokumenata u okviru veb aplikacija.

6. PRIKAZ SISTEMA

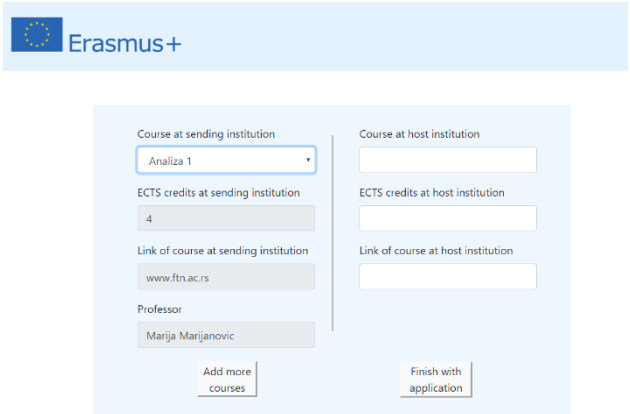
Pokretanje klijentske aplikacije vrši se iz browsera na računaru korisnika sistema.

6.1 Prijava na sistem i pregled funkcionalnosti

Rad sa aplikacijom korisnik započinje prijavljivanjem na sistem unošenjem korisničkog imena i lozinke. Na osnovu uloge korisnika otvara se početna stranica sa glavnim prikazom funkcionalnosti koje korisnik može da obavi.

6.2 Formiranje novog zahteva studenta

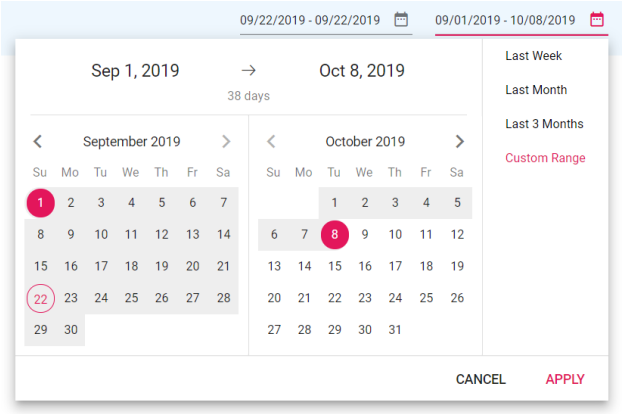
Osnovni podaci o studentu se sami popunjavaju, a zatim se unose podaci o stranoj instituciji na koju student konkuriše. Student može da dodaje proizvoljan broj kurseva koje želi da sluša tokom razmene (slika 6.1).



Slika 6.1 *Dodavanje kurseva koje student želi da sluša*

6.3 Pregled i filtriranje postojećih zahteva

Pregled istorije zahteva dostupan je svim korisnicima. Funkcionalnost sortiranja omogućena je po svim poljima u rastućem i opadajućem redosledu. Filtriranje podataka je po atributima *application created*, *status changed* i *status*. Filtriranje po datumu omogućeno je odabirom vremenskog intervala (slika 6.2).



Slika 6.2 *Prikaz odabira intervala za filtriranje po datumu*

6.4 Generisanje podloge za potpisivanje

Poslednji korak u procesu konkurisanja studenta na studije u inostranstvu jeste potpisivanje podloge za akademsko priznavanje perioda mobilnosti (slika 6.3).

Student: Jelena Ilic (index: SW42-2014)
Study programme: Softversko inzenjerstvo
Year of study: 4
Host university: University of Jaen

ERASMUS COURSES:

	Course at sending institution	ECTS credits to be recognised by the Sending Institution	Course at host institution	ECTS credits at the Host Institution
1	Masinsko ucenje	6	Machine learning	4
2	Analiza 1	4	Math	4

Head of study programme:

(Djordje Popovic)

Coordinator:

(Milan Petrovic)

Slika 6.3 *Izgled podloge za potpisivanje*

7. ZAKLJUČAK

Kako bi se olakšao osnovni proces konkurisanja na program mobilnosti u radu je opisana procedura podnošenja zahteva na Fakultetu tehničkih nauka u Novom Sadu, odnosno podrška za uvođenje automatizacije procesa podnošenja zahteva za studije u inostranstvu.

Opisan je način funkcionisanja Erasmus+ programa. Zatim je bilo reči o zahtevima sistema iz ugla korisnika. Navedene su korišćene tehnologije a kasnije i implementacija određenih delova sistema. U radu je dat opis modela podataka sistema. Takođe opisana je i podrška za vođenje procedure zahteva studenata putem modela poslovnog procesa. U poslednjem poglavlju, čitalac stiče sliku o celokupnom izgledu aplikacije iz ugla posmatranja korisnika.

Kako trenutna implementacija sistema podržava postupak kreiranja zahteva studenata samo na Fakultetu tehničkih nauka, dalje unapređenje aplikacije bilo bi proširenje sistema na nivo univerziteta. Proširivanjem sistema na veći broj fakulteta znatno bi povećao broj korisnika koji pristupaju sistemu. Ovo proširenje pre svega bi dovelo do znatnog povećanja broja poslatih e-mail poruka. Iz ovog razloga, trebalo bi razmisliti o korišćenju eksternih servisa koji pružaju API za razmenu e-pošte.

8. LITERATURA

[1] Ptak A, “Business -university cooperation in Europe” The educational program for 2014-2020 - key actions providing business cooperation with higher education institutions, Czestochowa University of Technology, 2012

[2] Ken Arnold, James Gosling, David Holmes, “The Java programming language”, Addison Wesley Professional, 17. avgust 2005.

[3] Pivotal „Spring“, Dostupno na: https://spring.io/, pristupljeno: 15. septembar 2019.

[4] The Apache Software Foundation “Apache Maven”, dostupno na: https://maven.apache.org, pristupljeno: 16. septembar 2019.

[5] „Angular”, dostupno na: https://angular.io/docs, pristupljeno: 16. septembar 2019.

[6] TechTarget “MySQL”, dostupno na: https://searchoracle.techtarget.com/definition/MySQL, pristupljeno: 16. septembar 2019.

[7] Alex Rodriguez, “RESTful Web services: The basics”, IBM, 6. novembar 2008.

[8] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, “Hypertext Transfer Protocol -- HTTP/1.1” - Introduction, jun 1999.

[9] M. B. Jones , B. Campbell , Ping Identity , C. Mortimore, “JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants, 29. mart 2013.

Kratka biografija:

Jelena Ilić rođena je 21. decembra 1995. godine u Novom Sadu, gde je 2010. godine završila osnovnu školu „Prva vojvođanska brigada“. Iste godine upisuje srednju ekonomsku školu „Svetozar Miletić“ u Novom Sadu, smer Ekonomski tehničar, koju završava 2014. godine. Iz osnovne i srednje škole izlazi kao nosilac Vukove diplome. Fakultet tehničkih nauka upisuje 2014. godine, smer Softversko inženjerstvo i informacione tehnologije. Polaže sve ispite predviđene planom i programom, nakon čega pristupa izradi diplomskog rada na temu „Generisanje alarma u sistemu za upravljanje logovima“. Zvanje diplomirani inženjer elektrotehnike i računarstva stiče 04.09.2018. godine sa prosečnom ocenom 9.61. Iste godine upisuje master akademske studije na smeru Softversko inženjerstvo i informacione tehnologije, odsek Elektronsko poslovanje. Polaže sve ispite master studija sa prosečnom ocenom 9.86.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE06Simic

SEMANTIČKI MODEL OBRAZOVNIH KOMPETENCIJA IT STRUČNJAKA

SEMANTIC MODEL OF EDUCATIONAL COMPETENCIES OF IT EXPERTS

Nevena Simić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je predstavljena ontologija obrazovnih kompetencija IT stručnjaka, koja treba da opiše znanja i veštine IT stručnjaka stečene kroz formalno i neformalno obrazovanje. Na osnovu ovog formalnog modela, kreiran je upitnik i izvršeno istraživanje kompetencija zaposlenih u IT kompanijama u Novom Sadu. Takođe, rezultati istraživanja su prikazani i obrazloženi, i dat je zaključak rada.*

Ključne reči: *Istraživanje; Formalno; Neformalno; Zaposleni*

Abstract – *This paper presents the ontology of educational competencies of IT experts, which should describe the knowledge and skills of IT experts acquired through formal and non-formal education. Based on this formal model, a questionnaire was created and the competencies of employees in IT companies in Novi Sad were examined. Explained results of the research are also presented and the conclusion of the paper is given.*

Keywords: *Research; Formally; Informally; Empolyee*

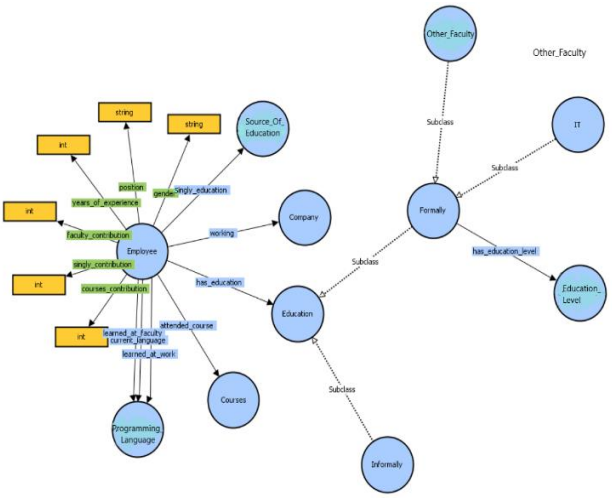
1. UVOD

Zadatak ovog rada je kreirati ontologiju, kao i na osnovu pomenute ontologije, uraditi istraživanje o formalnom i neformalnom obrazovanju zaposlenih u IT kompanijama u Novom Sadu. Biće predstavljena ontologija, definicija ontologije, jezik za ontologiju, kao i okruženje u kome se kreira ontologija. Pitanja na koja su odgovarali zaposleni u razvoju softvera, kao i zaposleni u ljudskim resursima takođe će biti izložena u radu, kao i grafici sa analizom.

2. ONTOLOGIJA

Prema [1] ontologija u računarstvu predstavlja rečnik (model podataka, ili informacioni model) u kojem se definišu osnovni koncepti koji će se razmatrati u određenom domenu. Može se sastojati od klasa, njihovih svojstava, ili akcija koje je moguće izvršiti u datom domenu. Na slici 1 predstavljena je ontologija kompetencija IT stručnjaka u obliku grafa, a to je omogućeno dodavanjem *plugin*-a VOWL u *Protégé* okruženje, koje služi za kreiranje ontologija.

Skicirano je sedam klasa: *Employee, Company, Education, Courses, Programming_Language, Source_Of_Education i Education_Level*, gde klasa *Education* ima dve potklase: *Formally* i *Informally*, jer obrazovanje zaposlenog može biti formalno ili neformalno. Klasa *Employee* ima svojstva objekata:



Slika 1 - *Ontologija*

- ***singly_education*** povezano sa klasom *Source_Of_Education*,
- ***working*** povezan sa klasom *Company*,
- ***has_education*** povezano sa klasom *Education* gde obrazovanje može biti formalno (IT formalno obrazovan ili formalno obrazovan u nekoj drugoj struci) ili neformalno, gde u slučaju formalnog, postoji svojstvo *has_education_level* povezano sa klasom *Education_Level* jer će samo formalno obrazovane osobe moći da imaju nivo obrazovanja: student, završene osnovne studije, završene master studije ili student doktorskih studija.
- ***attended_course*** povezano sa klasom *Courses*,
- tri svojstva koja su povezana sa klasom *Programming_Language*: ***learned_at_faculty, current_language*** i ***learned_at_work***.

Takođe, klasa *Employee* ima i svojstva podataka: *gender, position, years_of_experience, faculty_contribution, singly_contribution* i *courses_contribution***.**

3. SPISAK PITANJA KOJA SU POPUNJAVALI ISPITANICI ZAPOSLENI U RAZVOJU SOFTVERA I LJUDSKIM RESURSIMA

U ovom poglavlju će biti predstavljen spisak pitanja koja su popunjavali ispitanici zaposleni u razvoju softvera.

Rezultati i analiza skupljenih 80 odgovora iz 9 kompanija, kao i skupljeni odgovori ispitanika iz ljudskih resursa, biće predstavljeni u poglavlju: „Analiza rezultata". Ispitanici su odgovarali na sledećih 14 pitanja:

1. Pol?
2. Na kojoj ste poziciji u firmi?
3. Koliko godina iskustva imate u struci?

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

4. U kom programskom jeziku (jezicima) trenutno radite u firmi, i u kom *framework*-u?

5. Da li imate formalno IT visoko obrazovanje na nekom od studijskih programa na kojima se izučavaju IT tehnologije (npr. studije računarstva)? Pod formalno obrazovanim u ovoj anketi se smatraju i studenti završnih godina računarskih fakulteta. Ako je odgovor DA, upišite naziv fakulteta.

6. Ako nemate formalno IT visoko obrazovanje, da li imate formalno visoko obrazovanje u nekoj drugoj struci (da li ste završili neki drugi fakultet koji nije u vezi sa informacionim tehnologijama)? Ako je odgovor DA, upišite naziv fakulteta.

7. Navedite kakvo je Vaše formalno visoko obrazovanje?

8. Da li ste se neformalno obrazovali u IT struci kroz organizovane obuke (klasični i onlajn kursevi, radionice,...)? Ako jeste, molimo opišite koje ste sve obuke pohađali.

9. Navedite iz kojih izvora se samostalno obrazujete?

10. Koliko su Vašem znanju doprinele organizovane obuke? Napisati procentualno otprilike.

11. Koliko je Vašem znanju doprineo fakultet? Napisati procentualno otprilike, ukoliko ste formalno obrazovani u struci.

12. Koliko je Vašem znanju doprineo samostalni rad (izučavanje knjiga, foruma, uputstava,...)? Napisati procentualno otprilike.

13. Navedite najviše 3 programska jezika i *framework*-a sa kojima radite, a da ste ih naučili kroz formalno obrazovanje?

14. Navedite najviše 3 programska jezika i *framework*-a sa kojima radite, a da ste ih naučili kroz neformalno obrazovanje?

Spisak pitanja koja su popunjavali ispitanici zaposleni u ljudskim resursima biće predstavljen ispod. Ispitanici su odgovorili na sledećih 6 pitanja:

1. Broj zaposlenih koji učestvuju u razvoju softvera (programeri, testeri, sistem administratori,…)? Formalno obrazovanje podrazumeva završen IT fakultet ili je zaposleni pri kraju studija, dok neformalno obrazovanje podrazumeva da zaposleni nema završen IT fakultet. Ukoliko za neko polje nemate podatak, ostavite polje prazno.

2. Kroz kakve vidove učenja se zaposleni obrazuju na radnom mestu?

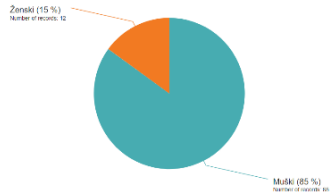
3. Da li firma zaposlene usmerava na formalno ili neformalno obrazovanje (navesti razlog)?

4. Kakva je struktura kandidata koji se prijavljuju za posao, da li su formalno ili neformalno obrazovani? Molimo da navedete koliki procenat kandidata je formalno obrazovan, a koliki procenat neformalno (otprilike)?

5. Molimo Vas da navedete razlike između formalno i neformalno obrazovanih kandidata za posao. Obrazložite kojoj od ove dve grupe kandidata dajete prednost i zašto? 6. Prokomentarišite kakav je tok i tempo napretka formalno obrazovanih zaposlenih u odnosu na neformalno obrazovane, i da li postoje razlike?

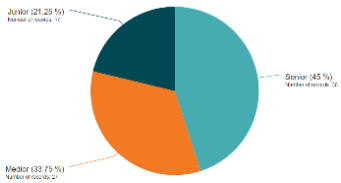
Na osnovu popunjene ankete od strane 80 osoba, analizom dobijenih rezultata, dolazi se do sledećih zaključaka (redom, na osnovu gore pomenutih pitanja):

1. Struktura zaposlenih u Novosadskim IT kompanijama je prikazana ispod (*slika 2*). Može se uočiti da je većina kandidata muškog pola (85%), dok strukturu osoba ženskog pola sačinjava 12 osoba (od ukupno 80), što je 15%.



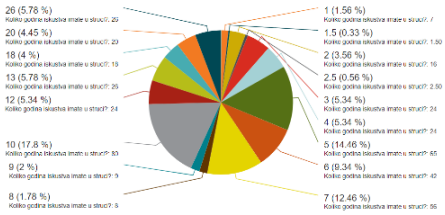
Slika 2 - Pol

2. Struktura pozicija na kojima zaposleni rade je prikazana na *slici 3*. Većina zaposlenih su seniori (45%), zatim mediori (33,75%) i juniori (21,25%). Iz toga se primećuje da u kompanijama većinom rade osobe sa najviše iskustva, dok je najmanje onih bez prethodnog iskustva.



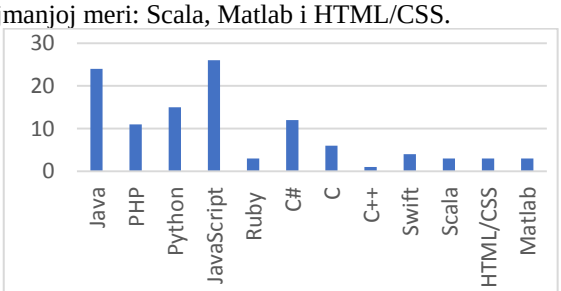
Slika 3 – Pozicija u kompaniji

3. Struktura kandidata po godinama iskustva je prikazana na *slici 4*. Vidi se da je najviše osoba koje imaju 10 godina radnog iskustva (17,8%), zatim 5 godina iskustva (14, 46%), a nakon toga 7 i 6 godina iskustva. Na osnovu ovih podataka, vidi se da najviše osoba ima između 5 i 10 godina radnog iskustva, što je podobno sa prethodno zaključenim (većina zaposlenih su seniori).



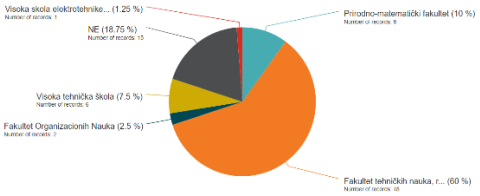
Slika 4 – Godine iskustva u struci

4. Struktura programskih jezika u kojima zaposleni rade je prikazana na *slici 5*. Može se primetiti da najviše zaposlenih rade programske jezike: Java, Python i JavaScript, kao i okvire za pomenute jezike, dok su manje zastupljeni: PHP (Laravel okvir), C#, C, Swift i Ruby, a u najmanjoj meri: Scala, Matlab i HTML/CSS.



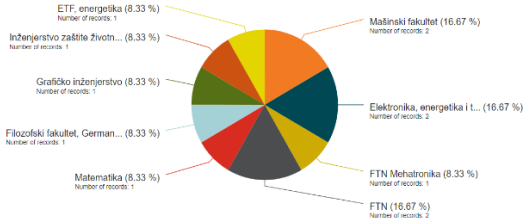
Slika 5 – Programski jezicu koje zaposleni koriste

5. Struktura zaposlenih koji imaju formalno IT obrazovanje prikazana je na *slici 6*. Najviše zaposlenih je formalno obrazovano u IT struci (81,25%), od kojih je većina završila Fakultet tehničkih nauka (60%), zatim Prirodno matematički fakultet (10%) i Visoku tehničku školu (8,75%).



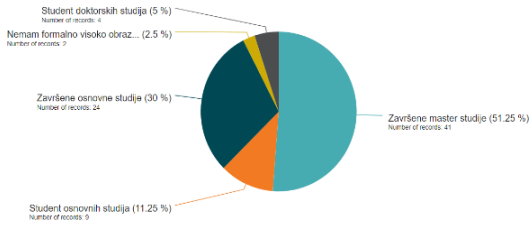
Slika 6 – Zaposleni koji su formalno obrazovani u struci

6. Struktura zaposlenih koji nemaju formalno IT obrazovanje, a imaju formalno obrazovanje u nekoj drugoj struci, prikazana je na *slici 7*. Uviđa se mešovita struktura osoba koje nisu formalno obrazovane u IT struci. Dolazi se do zaključka da je većina osoba studirala neku od prirodnih nauka.



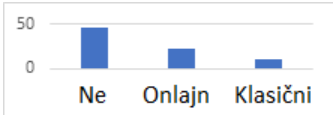
Slika 7 - Zaposleni koji nisu formalno obrazovani u struci

7. Struktura zaposlenih po nivou završenih studija je prikazana na slici 8. Vidi se da je većina zaposlenih formalno obrazovana (97,5%), od kojih je najviše osoba završilo master studije (51,25%), zatim osnovne studije (30%), i osoba koje još uvek studiraju je 11,25%. Uviđa se da je mali procenat osoba koje nisu formalno obrazovane (2,5%).



Slika 8 – Nivo završenih studija

8. Struktura obrazovanja kroz organizovane kurseve, radionice itd. je prikazana na slici 9. Može se videti da se većina osoba nije obrazovala kroz obuke, kurseve itd. (preko 50%), dok osobe koje jesu, većinom su pohađale onlajn kurseve-

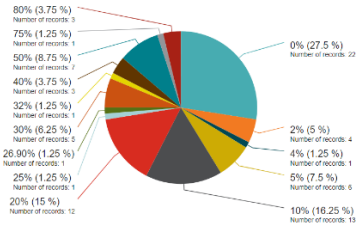


Slika 9 – Obrazovanje kroz kurseve

9. Struktura zaposlenih po izvoru iz kog se samostalno obrazuju je prikazana u tabeli. U tabeli se vidi da se najveći broj zaposlenih obrazuje iz knjiga, foruma i tutorijala (73 osobe od 80).

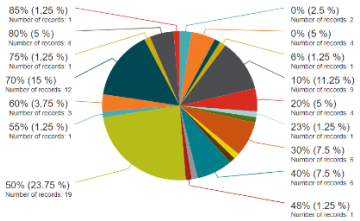
Naveditи iz kojih izvora se samostalno obrazujete	Broj zaposlenih
Online uputstva (tutorijali)	73
Knjige	73
Samo online uputstva	5
Samo forumi (tutorijali)	2

10. *Slika 10* prikazuje koliko su znanju doprinele organizovane obuke. Može se primetiti da nema mnogo doprinosa, jer je većina zaposlenih rekla da im uopšte nije doprinelo (27,5% osoba), dok je 16.25% njih reklo da im je doprinelo 10%, a 15% osoba je reklo da im je doprinelo 20%. Može se izvući zaključak da organizovane obuke u većini slučajeva doprinose trenutnom znanju od 0% do 20%.



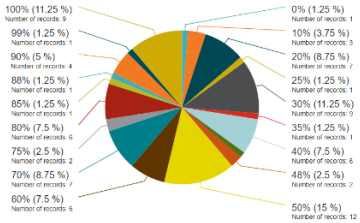
Slika 10 – Doprinos organizovanih obuka

11. Na *slici 11* je prikazano koliko je trenutnom znanju zaposlenih doprinelo znanje stečeno tokom studija. Uočava se da je najviše osoba reklo da im je fakultet doprineo 50%, a zatim 70%, što je ukupno 38,75% osoba koje su rekle da im je znanje sa fakulteta doprinelo u meri od 50% do 70%.



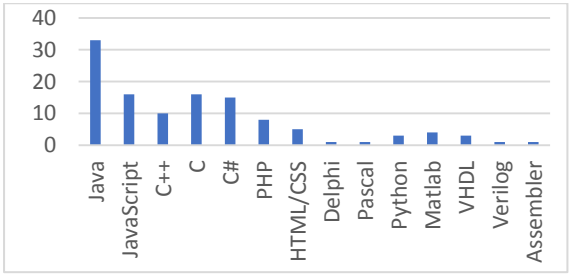
Slika 11 – Doprinos fakulteta

12. Na *slici 12* je prikazano koliko je trenutnom znanju doprineo samostalni rad. Može se videti da je najviše osoba izrazilo da im je samostalni rad doprineo u meri od 50% do 100%. 15% osoba je reklo da im je samostalni rad doprineo 50%, dok je 11,25% osoba reklo da im je doprineo 100%. Ostali procenat osoba se izjasnio mešovito.



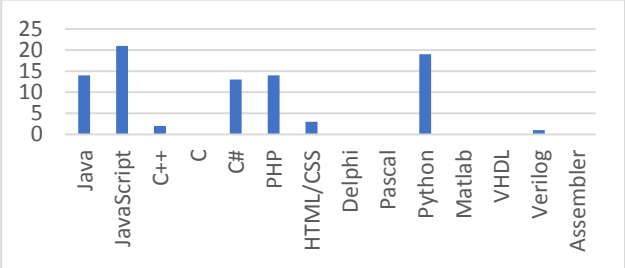
Slika 12 – Doprinos samostalnog rada

13. Struktura naučenih programskih jezika kroz formalno obrazovanje, s tim da ih osobe trenutno koriste u kompaniji, je prikazana na *slici 13*. Sa slike se uočava da je najveći broj osoba koje trenutno koriste navedene programske jezike, a naučili su ih kroz formalno obrazovanje: Java, JavaScript, C i C#, zatim u manjoj meri: C++, PHP, HTML/CSS, Matlab i Python.



Slika 13 – Formalno naučeni programski jezici

14. Struktura naučenih programskih jezika kroz neformalno obrazovanje s tim da ih osobe trenutno koriste u kompaniji, je prikazana na *slici 14*. Sa slike se uočava da je najveći broj osoba koje trenutno koriste navedene programske jezike, a da su ih naučili kroz neformalno obrazovanje: JavaScript i Python.



Slika 14 – Neformalno naučeni programski jezici

Gore navedena pitanja i prikazani rezultati su za osobe zaposlene u razvoju softvera, dok su ispod rezultati anketa koje su popunjavale osobe zaposlene u ljudskim resursima. Treba napomenuti da nisu svi zaposleni popunili anketu. Ukupan broj zaposlenih je 127 (u 9 kompanija iz kojih su zaposleni popunjavali ankete), od kojih ima ukupno 27 juniora, 39 mediora i 58 seniora, što se slaže sa analizom iz 2. pitanja kod kog se dobija da je najveći broj seniora među zaposlenima. Takođe, može se primetiti da je u svakoj od date 3 kategorije, najviše formalno obrazovanih osoba. Kod juniora je razlika formalno obrazovanih naspram neformalno obrazovanih 2:1, kod mediora 4:1 i kod seniora 7:1. Analizom rezultata o vidovima učenja kroz koje se zaposleni obrazuju na radnom mestu, većinom su dali odgovore da su to: onlajn kursevi i dokumentacija, učenje od iskusnijih kolega, dok su u manjoj meri rekli da su to i IT susreti, seminari, konferencije itd. Na pitanje da li kompanija zaposlene usmerava na formalno ili neformalno obrazovanje, osobe iz ljudskih resursa su odgovorile mešovito: formalno, zbog kredibiliteta prema stranim klijentima, zatim da im nije bitno da li imaju formalno ili neformalno obrazovanje, jer ako imaju znanje, nije bitno gde su zaposleni znanje stekli. Kod strukture kandidata koji se prijavljuju za posao u navedenih 9 kompanija, primećuje se da je prosečan broj (u procentima) kandidata koji se prijavljuju za posao, sledeći:

• Formalno obrazovani - 58%
• Neformalno obrazovani – 43%.

Zaključuje se da je veći procenat kandidata formalno obrazovan, ali razlika nije velika. Na pitanje da li postoje razlike između formalno i neformalno obrazovanih kandidata, i kojim osoba se daje prednost, 7 osoba (7 kompanija) je odgovorilo da prednost daju formalno obrazovanim kandidatima, dok je jedna osoba rekla da nije bitno da li je kandidat formalno ili neformalno obrazovan. Na pitanje toka i tempa napretka formalno obrazovanih u odnosu na neformalno obrazovane, i da li postoje razlike među njima, odgovori su većinom bili da formalno obrazovani kandidati brže uče, kao i da već imaju radne navike i predznanje, dok je to kod neformalno obrazovanih kandidata manje izraženo.

5. ZAKLJUČAK

U ovom radu predstavljena je ontologija obrazovnih kompetencija IT stručnjaka iz IT kompanija u Novom Sadu, gde su opisane njihove veštine, kao i znanja koja su stekli tokom formalnog ili neformalnog obrazovanja. Takođe, predstavljena je i anketa sa analizom rezultata. Nedostatak ove analize je broj kompanija, kao i broj osoba koje su popunjavale anketu. Kako raste broj kompanija i osoba, raste i tačnost dobijenih rezultata, stoga bi u budućnosti moglo da se uradi detaljnije istraživanje.

6. LITERATURA

[1] Diana Man, *Ontologies in Computer Science*, Didactica Mathematica, 2013.
[2] Tamara Đorđević, *Semantički veb*, Visoka ICT škola u Beogradu, 2012.
[3] Luka Gospodnetić, *Semantički veb*, Fakultet elektrotehnike i računarstva ZEMRIS, 2015.
[4] Tim Berners-Lee, James Hendler, Ora Lassila, *The Sematic Web*, Scientific American, 2001.
[5] Catherine Legg, *Ontologies on Semantic Web*, University of Waikato, Hamilton-Tauranga, New Zealand, 2010.

Kratka biografija:

Nevena Simić rođena je u Novom Sadu 1993. godine. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Semantički veb odbranila je 2019.god. kontakt: nevena_simic@outlook.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE08Ribar

DOPRINOS *PBR* PRISTUPA KVALITETU *glTF* MODELA

THE CONTRIBUTION OF *PBR* APPROACH TO A BETTER *glTF* MODEL RENDERING

Miloš Ribar, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – *Opisi modernih koncepata iz računarske grafike, njihov doprinos u industriji i u samom izgledu scene koju grafički softver daje kao rezultat rada, koristeći moderne tehnike renderinga.*

Ključne reči: *Računarska grafika, svetlosne jednačine, šejderi, rendering baziran na fizici, BRDF, glTF*

Abstract – *Descriptions of modern computer graphics concepts, their contributions to the industry and the look of the scene that the graphics software renders.*

Keywords: *Computer graphics, light equations, shaders, physically based rendering, BRDF, glTF*

1. UVOD

Pored filmske industrije, industrija video igara je jedna od najrazvijenijih grana industrije zabave u 21. veku. Budžeti za izradu proizvoda i jedne i druge grane industirje su multimilionski, i na njihovoj izradi je zaposleno više timova sa velikim brojem ljudi.

Glavni zahtev konzumenata multimedijalnog sadržaja je da on bude što realističniji i pruži konzumentima osećaj kao da su stvarno u filmskoj sceni koju gledaju, ili u video igri koju igraju. PBR je skup tehnika senčenja i renderovanja koja se bavi problemom realizma.

Ljudi su počeli ozbiljno da shvataju PBR još 1980. godine, kada se prvi put pojavio koncept rendering metode poznate kao ***ray tracing***. Ovaj koncept je bio prvi put predstavljen od strane američkog računarskog naučnika Turner Viteda (eng. ***J. Turner Whitted***) u njegovom naučnom radu: „Poboljšan model osvetljenja za osenčeni displej".

Godine 1981. i 1982. pojavilo se unapređenje u PBR pristupu renderinga u vidu usavršenog refleksionog modela. Kuk-Torens (eng. *Cook-Torrance*) refleksioni model je uveo pojam mikrofaseta i mikrofasetne refleksije u grafiku. Kompanija NVidia je početkom 2019-e godine izbacila novu seriju grafičkih kartica sa arhitekturom koja služi isključivo za ***ray tracing*** u realnom vremenu.

2. FOTOREALIZAM I *RAY TRACING*

Fotorealizam je proces stvaranja slike koristeći rendering mehanizme, tako da dobijena slika bude vrlo realistična.

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

Postoji više vrsta ray tracing algoritama i njihovih varijacija. ***Light-based*** algoritmi generišu zrake iz izvora svetlosti. Najčešći, ***eye-based*** funkcionišu po obrnutom principu, gde se zraci generišu iz kamere. Skoro svi fotorealistični rendering sistemi su zasnovani na korišćenju ray tracing algoritma i njegovih varijacija. Svaka varijacija algoritma mora da poštuje simuliranje skupa fenomena i prisutnost određenih objekata u sceni.

2.1. KAMERA

Krucijalni problem koji kamera treba da reši je da odredi koje će boje biti svaki piksel na slici. Zadatak kamere je da uzme jedan piksel na slici i da izgeneriše zrak koji prolazi kroz taj piksel. Zrak može, a ne mora da se sudari sa nekim objektom na svom putu, može se prelomiti kroz taj objekat i nastaviti svoju putanju, a može se i reflektovati od tog objekta. Ako se zrak sudari sa objektom, računa se incidentno svetlo (ukupno svetlo koje pada na površinu objekta) koje doprinosi boji posmatranog piksela. Zraci se generišu iz kamere, i oni se propagiraju po sceni.

2.2. PRESEK ZRAKA I OBJEKTA

Presek zraka sa objektom na sceni se mora precizno pronaći. Na tački preseka zraka sa objektom treba odrediti njegova svojstva u toj tački kao što su normala površine ili materijal od kog je taj objekat sačinjen. Svaki put kada kamera izgeneriše zrak, zadatak renderera je da odredi da li postoji presek sa objektom i gde se on tačno nalazi.
Zbog ovoga, ***ray tracing*** algoritam najčešće traje $O(I \log N)$, gde je I broj piksela, a N broj objekata na sceni. Presek zraka i objekta se može naći pomoću kombinacije jednačine zraka $r(t) = o + td$, implicitne funkcije $F(x, y, z) = 0$ i ubacivanjem jednačine objekta u implicitnu funkciju koja sadrži nepoznatu t. Za sferu se dobija jednačina koja ima oblik kao jednačina ispod.

$$(o_x + td_x)^2 + (o_y + td_y)^2 + (o_z + td_z)^2 - r^2 = 0.$$

Ako ne postoje realni koreni jednačine, zrak se ne sudara sa sferom. Ako pak postoje, najmanji realni koren će biti tačka sudaranja zraka i objekta.

2.3. SVETLOSNI IZVORI I DISTRIBUCIJA SVETLOSTI

Presek zraka i objekta daje fragment koji treba osenčiti kao i svojstva objekta u toj tački. Potrebno je saznati kolika je količina svetlosti koja napušta fragment objekta. Ovo se može saznati tako što se uzmu u obzir svi svetlosni zraci koji padaju na fragment. Ovi proračuni

zahtevaju uvođenje radiometrijskih i geometrijskih formula distribucije svetlosti. Kod izračunavanja uticaja svetlosti na izgled objekta treba uzeti u obzir opadanje energije svetlosti sa rastojanjem svetlosnog izvora i objekta, i nagib objekta u odnosu na svetlosni izvor.

Snaga svetlosti koja pristiže na površinu objekta trpi prigušenje pri putovanju svetlosnog zraka kroz prostor. Ovo je opisano funkcijama (1) i (2).

$$f_{att} = 1/r^2 \quad (1)$$

$$f_{att} = 1/(k_c + k_l d + k_q d^2) \quad (2)$$

Ako je diferencijalna površina dA nagnuta za neki ugao θ u odnosu na normalu i zrak svetlosti, snaga kojom svetlost utiče na taj deo površine je skaliran koeficijentom $cos\theta$.

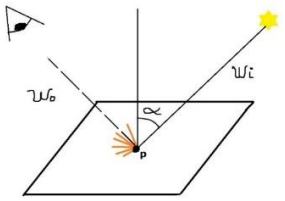
2.3. VIDLJIVOST

Između posmatranog objekta i svetlosnog zraka može da se nađe prepreka i potrebno je testirati da li je zrak pogodio prepreku i da li ta prepreka stvara senku na objektu.

Svaki svetlosni zrak iz izvora svetlosti doprinosi osvetljenju objekta samo ako je put od svetlosnog izvora do objekta neometan. Ovo se može proveriti tako što se napravi jedan zrak sa izvorom u tački na posmatranoj površini, sa pravcem i smerom ka svim izvorima svetlosti. Ovi zraci se zovu ***zraci senke***. Parametar ***t*** koji se odnosi na neki presek se uporedi sa parametrom ***t*** zraka koji putuje ka svetlosnom izvoru. Ako svetlost nije blokirana na ovom putu, njen uticaj se uključuje u dalje proračune.

2.3. ODBIJANJE SVETLOSTI OD POVRŠINE OBJEKTA

Svaki objekat mora da sadrži opis kako se svetlost ponaša posle sudaranja sa njim. Najbitnije svojstvo je rasipanje incidentne svetlosti. Potreban podatak je količina svetlosne energije koja se odbija duž snopa svetlosti koji se sudara sa posmatranom tačkom na objektu. Ova situacija je prikazana slikom 1.

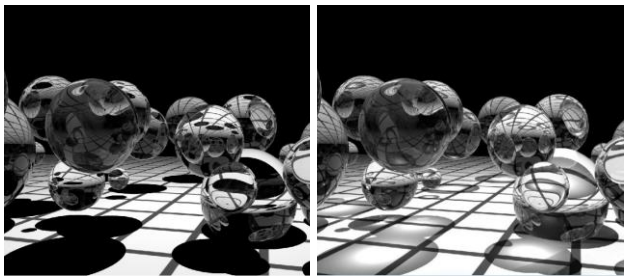


Slika 1: *Rasipanje svetlosti posle udarca o tačku na površini objekta.*

Potrebno je odrediti energiju zraka koji prati pravac kamere ω_0. Funkcija koja opisuje raspršivanje svetlosti se naziva funkcija distribucije bidirekcione refleksije (BRDF) koja se može zapisati u obliku $f_r(p, \omega_i, \omega_0)$.

2.3. INDIREKTAN TRANSPORT SVETLOSTI

Pošto svetlost može da stigne na površinu nakon odbijanja od neke druge objekte u sceni, važno je pratiti svaki zrak sve dok on ne izgubi svoju energiju. Ako zrak iz kamere udara u ogledalo, dolazi do refleksije svetlosti u odnosu na normallu površine ogledala i upadnog ugla svetlosnog zraka. U ovoj situaciji se algoritam može pozvati rekurzivno. Ova tehnika se može iskoristiti i za transparentne objekte, gde se umesto reflektovanog zraka koristi prelomljeni. Svaka varijanta algoritma daje različite rezultate, gde se to najbolje vidi u senkama transparentnih objekata. Slika 2 predstavlja odnos između ***photon mapping*** i Vajtedovog ***ray tracing*** algoritma, gde ***photon mapping*** koristi ***ray tracing*** u drugom prolazu.



[1] Slika 2: *Leva slika predstavlja Vajtedovu metodu **ray tracing** algoritma. Desna slika predstavlja **photon mapping** algoritam.*

3. RENDERING BAZIRAN NA FIZICI

Rendering zasnovan na fizici je skup rendering tehnika koje su zasnovane na istoj teoriji, koje za cilj imaju simulaciju izgleda stvarnog sveta. Ovakvi rendering sistemi aproksimiraju ponašanje svetlosti u realnom svetu oslanjajući se na svetlosne fizičke formule iz oblasti radiologije, gde je glavni fokus stavljen na interakciju svetlosti sa raznim materijalima. Materijali nose sa sobom razne podatke u jednom fragmentu, o tome koliko su hrapavi, da li se u tom fragmentu nalaze udubljenja ili oštećenja, kakva je boja materijala i kakav je odsjaj u tom fragmentu. Postoji više vrsta tekstura koje doprinose realnosti izgleda objekata. Ove teksture prave umetnici.

Da bi jedan PBR svetlosni model bio baziran na fizici, mora da poštuje sledeće uslove:

1. Baziran je na modelu mikrofasetnih površina.
2. Mora poštovati zakone očuvanja energije.
3. Mora koristiti aproksimaciju BRDF jednačine kao rešenje svetlosne formule.

3.1. MIKROFASETNI MODEL

Svaki grafički objekat je presvučen nekim materijalom gde taj materijal može biti grub u većoj ili manjoj meri. Grubost materijala ogleda se u sitnim izbočinama na njegovoj površini. Te izbočine se ponašaju kao sićušna ogledala i nisu vidljive golim okom. Pogledati sliku 3. Haotičan raspored mikrofaseta označava jako grub materijal, a grubost utiče na haotičnu refleksiju svetlosnih zraka, što dovodi do većeg područja spekularne refleksije.

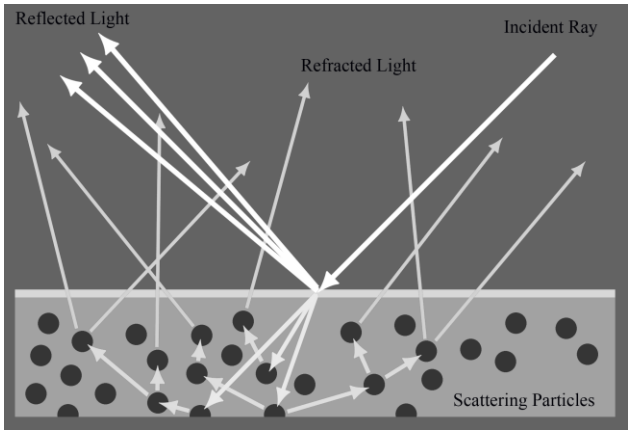
[2] Slika 3: *Mikrofaseti na grubim i glatkim površinama.*

Vrednost grubosti površine se opisuje statističkim parameterom grubosti (roughness). Na osnovu ovog parametra se može odrediti koliko je mikrofaseta orijentisano ka vektoru poluputanje $\vec{h}$.

3.2. OČUVANJE ENERGIJE

Tehnike senčenja zasnovane na fizici uključuju i zakon održanja energije. Jačina izlazeće svetlosti nikada ne sme da pređe jačinu ulazne svetlosti, i da bi neki sistem bio fizički baziran, ovo je važan uslov. Ovo se ne odnosi na objekte koji emituju svetlost.

Pretpostavka je da je svaka površina napravljena od dva sloja. Primer se može videti na slici 4. U momentu kada svetlosni zrak padne na površinu objekta, emitujuća svetlost objekta deli se na reflektovanu i refraktovanu svetlost. Refraktovana svetlost ulazi u međuprostor između dva sloja materijala i sudara se sa česticama koje čine materijal, gde potencijalno ostaje apsorbovana.



[2] Slika 4: *Dvoslojna arhitektura materijala.*

Očuvanje energije je obezbeđeno tako što se prvo izračuna uticaj spekularne refleksije, a potom se difuzna izvede iz spekularne.

3.3. FUNKCIJA DISTRIBUCIJE BIDIREKCIONE REFLEKSIJE

Aproksimira izgled neprozirne površine na osnovu svojstava materijala kojom je presvučena. Kao ulazne parametre uzima vektore ω_i, ω_0 i koeficijent grubosti mikrofaseta materijala a. Svaka BRDF funkcija mora da ima izlaz ne veći od 1.0.

Rendereri koji rade u realnom vremenu najčešće koriste ***Cook-Torrance BRDF*** koja je oblika u nastavku teskta. Varijabla *c* predstavlja albedo materijala, ***D*** predstavlja funkciju raspodele mikrofaseta, ***G*** je funkcija raspodele geometrijske okluzije, a ***F*** je Frenelova jednačina.

$$f_r = k_d f_{lambert} + k_s f_{cook-torrance},$$

$$f_{lambert} = \frac{c}{\pi}$$

$$f_{cook-torrance} = \frac{DFG}{4(\omega_0 * n)(\omega_i * n)}$$

4. REALISTIČNO OSVETLJENJE U PBR ENDŽINIMA

Najznačajniji deo u prezentaciji jedne scene u video igrama i filmovima, predstavlja osvetljenje. Ono mora biti postavljeno tačno onako kako treba da ne bi došlo do lošeg mešanja boja i dobijanja nepregledne scene.

Primena PBR tehnika nije toliko komplikovana ako postoji nekoliko izvora svetlosti. Međutim, u realnosti, okruženje i objekti u okruženju predstavljaju dodatni izvor svetlosti. U tom slučaju, ne može se znati unapred koliko izvora svetlosti postoji na sceni. Tehnike osvetljenja koje rešavaju ovakav slučaj zovu se tehnike osvetljenja zasnovanog na slici ili *Image Based Lighting*.

IBL tehnike osvetljenja oslanjaju se na prostor u kom se objekti nalaze, gde je taj prostor ustvari mapa prostora (cubemap). Svaki teksel mape okruženja može biti posmatran kao jedan izvor svetlosti. Da bi integral izračunao efikasnije, većinu stvari treba izračunati unapred. Rendering jednačina se jednostavno rastavi na spekularni i difuzni integral.

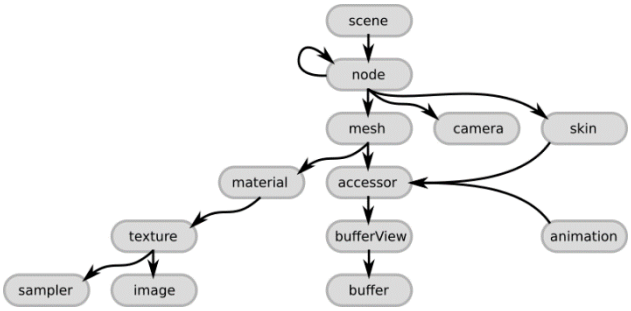
$$L_0(p, \omega_0) = k_d \frac{c}{\pi} \int_\Omega L_i(p, \omega_i) \cos\theta \, d\omega_i$$

$$L_0(p, \omega_0) = \int_\Omega \frac{DFG}{4(\omega_0 * n)(\omega_i * n)} L_i(p, \omega_i) \cos\theta \, d\omega_i$$

5. *glTF* FORMAT PODATAKA

Neobrađeni 3D podaci mogu doći u raznim formatima, kao što su .obj, .pli, .stl datoteke. Ovi formati datoteka ne sadrže informacije o tome kako bi jedan 3D model, ili čitava scena, trebala biti prikazana na ekranu. Struktura scene se mora analizirati, a geometrijski podaci se moraju pretvoriti u format koji grafička aplikacija podržava. svaka aplikacija mora da implementira svoj importer, louder i konvertere za sve formate datoteka koje će podržavati. Ovo je dosta veliki problem za programere.

Format podataka Khronos grupe, glTF pokušava da reši gore pomenuti problem tako što će standardizovati prikaz 3D sadržaja u klijentskim aplikacijama. On nije još jedan format podataka, već je definicija transmisionog podatka za 3D scene.



[3] Slika 5: *Povezanost delova glTF objekta*

6. VULKAN GRAFIČKA BIBLIOTEKA

Vulkan je retained biblioteka, koja sav posao oko kreiranja grafičkih komponenti ostavlja programeru. Najmanji detalj koji se tiče biblioteke i grafičkog pajplajna programer mora sam da kreira. Većina posla u korišćenju ove biblioteke svodi se na inicijalizaciju struktura i pozivanje različitih funkcija koje će komunicirati direktno sa drajverom, preko različitih slojeva biblioteke.

7. LITERATURA

[1] Matt Pharr, Wenzel Jakob, Greg Humphreys, Physically Based Rendering: From Theory to Implementation, 2018

[2] Jeremy Romanovski, Let's Get Physical: https://www.jeremyromanowski.com/blog/2015/11/20/lets-get-physical-part-1-of-3

[3] glTF github repozitorijum https://github.com/KhronosGroup/glTF

Kratka biografija:

Miloš Ribar rođen je u Beogradu 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Doprinos *PBR* pristupa kvalitetu *glTF* modela odbranio je 2019.god.
kontakt:
milosribar@yahoo.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE09Joksimovic

ODREĐIVANJE KVALITETA POJEDINAČNIH JELA IZ RECENZIJA RESTORANA UPOTREBOM SENTIMENT ANALIZE

DETERMINATION OF QUALITY OF INDIVIDUAL MENU ITEMS FROM RESTAURANT REVIEWS USING SENTIMENT ANALYSIS

Marija Joksimović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu predstavljeno je jedno rešenje za izdvajanje delova teksta iz korisničkih recenzija koji predstavljaju nosilac sentimenta za određeno jelo i klasifikaciju tih delova teksta prema osećanju koje je u njima iskazano na pozitivno, negativno ili neutralno korišćenjem metoda mašinskog učenja. Segmenti teksta recenzija koji sadrže pominjanje hrane su formirani na osnovu leksičkih veza između reči i na njih su primenjene određene tehnike pretprocesiranja. Potom je izvršena sentiment analiza upotrebom nekoliko modela mašinskog učenja. Svi podaci su prikupljeni sa sajta Donesi.com i ručno anotirani. Korišćeni modeli su evaluirani.*

Ključne reči: *analiza teksta, klasifikacija teksta, procesiranje prirodnog jezika, sentiment analiza, recenzije restorana*

Abstract – *This paper presents one approach for extraction of parts of restaurant reviews which contain information about opinion of certain menu item and classification of those segments by expressed sentiment as positive, negative or neutral using several machine learning algorithms. Text segments which contain food mentions were generated using lexical relationships between words in reviews and several preprocessing techniques were applied. Afterwards, sentiment analysis was done using several machine learning models. Data was acquired from the website Donesi.com and manually annotated. All used models were evaluated.*

Keywords: *text mining, text classification, natural language processing, sentiment analysis, restaurant reviews*

1. UVOD

Ljudi prilikom donošenja odluka, posebno kada je u pitanju kupovina određenog proizvoda ili usluge, konsultuju mišljenje drugih ljudi pre nego što donesu svoj konačni sud.

Pre ekspanzije interneta pojedinci su bili ograničeni na porodicu i prijatelje ukoliko im je bila potrebna pomoć prilikom donošenja odluke, a kompanije su bile primorane da sprovode različite studije i ankete kako bi prikupili podatke o zadovoljstvu svojih klijenata.

Danas to više nije slučaj. Informacije od značaja su sadržane u korisničkim recenzijama i komentarima na veb stranicama onih koji proizvod nude i najčešće su javno dostupni. Jedan slučaj prethodno opisanog scenarija jesu recenzije na sajtovima za preporuku restorana.

Većina takvih sajtova korisnicima omogućuje pretragu najbolje rangiranih restorana iz određene kategorije, ali se retko sreće slučaj preporuke određenih stavki jelovnika u okviru restorana.

Tema ovog rada jeste izdvajanje delova teksta iz korisničkih recenzija koji predstavljaju nosilac sentimenta za određeno jelo i klasifikacija tih delova teksta prema osećanju koje je u njima iskazano na pozitivno, negativno ili neutralno korišćenjem metoda mašinskog učenja. Obzirom da srpski jezik spada u leksički složene jezike i pošto je čest je slučaj da se u okviru iste recenzije spominje više jela gde svako od njih može biti opisano u drugačijem svetlu, najveći izazov jeste izdvojiti okolinu oko datog pominjanja hrane tako da obuhvati samo jedno jelo i kontekst u okviru kog se ono spominje.

Čitav proces dodatno otežava činjenica da recenzije korisnika spadaju u kraće, neformalne tekstove, u kojima se često koriste žargonski izrazi i sadrže šum. Analiza sentimenta kao vid klasifikacije teksta je poslednjih nekoliko godina sve češća tema naučnih istraživanja.

Osnovne razlike leže u izboru domena, definisanju osnovne jedinice dokumenta za koju će se određivati sentiment, izboru tehnika pretprocesiranja teksta i algoritama mašinskog učenja koji će se primeniti.

Tekstovi sadržani u domenima mogu biti duži i često formalni tekstovi poput novinskih članaka, književnih radova i blogova, ili kraći, neformalni tekstovi, često puni šuma poput online recenzija i postova sa društvenih mreža.

Skup podataka formiran je od recenzija preuzetih sa sajta Donesi.com [1], jednog od najpopularnijih sajtova za naručivanje hrane preko interneta na teritoriji Srbije, Crne Gore i Bosne i Hercegovine.

Od svih recenzija koje su preuzete, nasumično su izdvojene recenzije pisane na srpskom jeziku koje će ući u test i trening skupove. Delovi teksta recenzija u kojima je iskazano mišljenje o nekom jelu su izdvojeni na osnovu leksičkih veza između reči upotrebom biblioteke za procesiranje prirodnih jezika (eng. *Natural Language Processing - NLP*) specijalizovane za srpski jezik i vršena je sentiment analiza na nivou entiteta.

Korišćena su ukupno četiri algoritma nadgledanog mašinskog učenja za klasifikaciju: Naive Bayes, SVM, Random Forest i logistička regresija.

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

2. METODOLOGIJA

U ovom poglavlju predstavljena je metodologija određivanja delova teksta recenzija u kojima je iskazano mišljenje o nekoj hrani i klasifikacija tih delova prema sentimentu koji nose na pozitivan, negativan ili neutralan. Detaljno će biti predstavljen čitav postupak sa svim koracima i alatima koji su implementirani i korišćeni tokom realizacije.

2.1. Prikupljanje podataka

Koliko je autor rada upoznat, ne postoji javno dostupan skup podataka koji sadrži recenzije restorana koje su pisane na srpskom jeziku. Svi neophodni podaci su preuzeti sa Donesi.com. Pošto sajt ne poseduje API za preuzimanje podataka napisan je specijalizovani web crawler u programskom jeziku Python. Za implementaciju crawler-a korišćena je biblioteka Selenium [2], a za parsiranje HTML stranica biblioteka BeautifulSoup4 [3]. Odlučeno je da se dobave svi podaci vezani za područje Srbije. Dobavljeno je ukupno 169486 recenzija za 1658 različita restorana.

2.2. Anotacija pominjanja hrane u recenzijama

Neophodno je bilo označiti tačna mesta u tekstu gde su se javila pominjanja hrane. Kako bi bile anotirane samo recenzije koje sadrže barem jedno pominjanje hrane u naslovu ili u konkretnom tekstu napravljen je jednostavan program. Recenzije su učitavane sukcesivno i prikazivane anotatoru koji je označavao da li trenutna recenzija sadrži pominjanje hrane ili ne. Ako je odgovor bio potvrdan, recenzija bi se sačuvala u novu kolekciju. Manuelno je anotirano 10000 recenzija u kojima su anotatori označavali svaku reč ili grupu reči koja predstavlja pominjanje hrane.

2.3. Proširivanje pominjanja hrane iskazanim mišljenjem

Tekstovi recenzija su prvo podeljeni u rečenice i svaka rečenica je tokenizovana. Za svaki token ubeležen je i njegov položaj u originalnom tekstu. Iako se za analizu sentimenta nije korišćena cela recenzija, POS tagovanje, izvlačenje povezanih reči i lemantizacija su izvršeni u ovom koraku kako bi ostala očuvana leksička veza između reči. Originalni tekst recenzije nije izmenjen, samo su zasebno sačuvani nizovi tokena, njihovih POS tagova i lema.

Korišćena je ReLDI [4-8] biblioteka namenjena za procesiranje tekstova na srpskom, hrvatskom i slovenačkom jeziku. Svi izlazi koje daje ReLDI biblioteka su u JSON formatu i nakon procesiranja su povezani sa odgovarajućim objektima recenzija koji su prethodno sačuvani. Svi tokeni koji pripadaju nekom pominjanu hrane su označeni parsiranjem rezultata anotiranja iz prethodnog koraka. Tabelarni prikaz izlaza procesiranja rečenice pomoću ReLDI prikazan je u tabeli 1.1.

Reči koje predstavljaju hranu ili imena jela su najčešće imenice. Imena hrane ili delova jela se sastoje iz jedne ili više imenskih reči kao što su : pizza, pasta sos, hleb, paradajz itd. Nazivi jela se mogu sastojati i iz drugih vrsta reči koje bliže opisuju sastojke ili poreklo određenog jela, poput „Bečka šnicla“ ili „piletina u slatko–kiselom Sečuan sosu“. Čest je slučaj da se u nazivima jela javi ime restorana, poput „The Saint Pizza“. Takođe, postoji dosta restorana strane kuhinje u kojima neka ili sva jela iz jelovnika nose ime na izvornom jeziku. Korisnici u recenzijama hranu spominju prema sopstvenom nahođenju. Nazivi jela su retko napisani onako kako su navedeni u jelovniku.

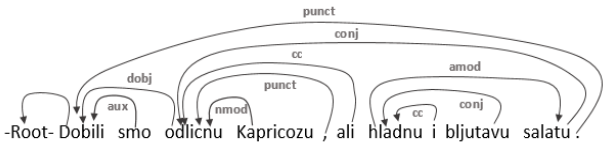
Najčešće se koriste skraćeni ili prevedeni nazivi jela (Quattro Formagi Pizza – Pica 4 sira) i fonetska transkripcija (Quattro Formagi Pizza – Kvatro formađi pica). Čest je i slučaj korišćenja kovanica i gramatički neispravnih oblika reči. Sve ovo za posledicu ima javljanje grešaka prilikom automatskog određivanja vrste reči za svaki token tokom POS tagovanja, lemantizacije i formiranja stabla zavisnosti.

Isto tako, korisnici često ne iskazuju mišljenje direktno o nekom jelu, nego sentiment iskazuju kritikom neke komponente tog jela, poput primera „Salata sa groznim, zagorelim pečurkama“. Glavni izazov u ovom koraku jeste definisati algoritam uparivanja i skupa dozvoljenih veza između reči unutar recenzije koje su povezane sa datim pominjanjem hrane tako da se uhvati sentiment vezan samo za jedno jelo.

Tabela 1. *Procesiranje rečenice pomoću ReLDI biblioteke*

ID	Token	POS tag	Lema	Dep-parse
1	Dobili	Vmp-pm	dobiti	0/root
2	smo	Var1p	biti	1/aux
3	odlicnu	Ncfsa	odlicna	1/dobj
4	Kapricozu	Npmsl	Kapricoz	3/nmod
5	,	Z	,	3/punct
6	ali	Cc	ali	3/cc
7	hladnu	Agofsay	hladan	10/amod
8	i	Cc	I	7/cc
9	bljutavu	Agpfsay	bljutav	7/conj
10	salatu	Ncfsa	salata	3/conj
11	.	Z	.	1/punct

Parsiranje zavisnosti se vrši na nivou rečenice. Parser zavisnosti iz biblioteke ReLDI [7] je sposoban da prepozna 39 različitih tipova veza. Svaka veza se sastoji iz dva tokena i naziva tipa veze. Jedan token iz veze predstavlja „glavu“ (eng. *head*), dok je drugi zavisni token u vezi. Jedan token iz rečenice nema svog para, predstavlja koren i označen je vezom „root“. Iako izlaz iz parsera zavisnosti jeste niz objekata veza, rekurzivnim obilaskom tog niza se dobija stablo zavisnosti tokena u rečenici. Svi tokeni osim korena se u vezama mogu javiti proizvoljan broj puta. Jedan primer izgleda stabla zavisnosti prikazan je na slici 1.



Slika 1. *Stablo zavisnosti*

Lista tipova veza koje su uzete u obzir je dobijena empirijski. Obično su korišćene veze *amod*, *admod*, i *nsubj*. Na taj način je jedino moguće proširiti pominjanje hrane sa jednom rečju ili frazom koja ga šire opisuje.

Zbog prirode problema odlučeno je da se koristi više veza kako bi bilo moguće obraditi i slučajeve gde je sentiment iskazan za određeni deo jela. Isprobani su različiti skupovi tipova veza i odlučeno je da se u finalnom skupu nađe osam : *advmod* (eng. *adverbial modifier*), *amod* (eng. *adjectival modifier*), *acl* (eng. *adjectival clause*), *nmod* (eng. *nominal modifier*), *nsubj* (eng. *nominal subject*), *neg* (eng. *negation modifier*), *nummod* (eng. *numeric modifier*), *dobj* (eng. *direct object*). Opise tipova veza je moguće pronaći na [10].

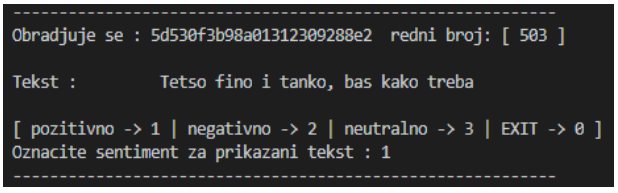
Za automatsko izvlačenje segmenata koji sadrže pominjanje hrane i mišljenje vezano za njih je napisan program. Obrađeno je svih 10000 anotiranih recenzija. Za svaku recenziju preuzet je skup svih pominjanja hrane. Svako pominjanje hrane je obrađeno zasebno. Inicijalni skup tokena koji predstavlja izlazni segment su činili tokeni datog pominjanja hrane. Provera koji se tokeni iz recenzije odnose na druge tokene segmenta obavljao rekurzivno. Za svaku vezu iz liste zavisnosti provereno je da li postoji veza gde su zadovoljene sledeće stavke :

- Neki token iz trenutnog skupa tokena koji čine segment ima ulogu glave
- Podređeni token se ne nalazi u trenutnom skupu
- Tip veze pripada skupu validnih tipova veza

Ukoliko su sva tri uslova zadovoljena token bi bio dodat u skup tokena za narednu obradu. Kada bi bile ispitane sve veze za trenutni skup tokena, ponovo bi bio pokrenut proces obrade, ali ovaj put ulaz bi bio skup generisan u prethodnoj iteraciji. Čitav proces bi se rekurzivno ponavljao dok se u jednoj iteraciji ni jedan token ne nađe u skupu za narednu obradu.
Nakon toga, skup tokena bi bio pretvoren u listu sortiranu rastuće prema poziciji tokena u tekstu. Iz originalnog teksta recenzije bio bi isečen deo između početka prvog tokena u listi i kraja poslednjeg. To je učinjeno kako bi ostao očuvan smisao. Tokeni i leme svih reči, bez obzira da li su pripadali izlaznom skupu tokena ili su naknadno dodati nakon isecanja, zasebno su sačuvani u novu kolekciju. Sačuvani su samo segmenti gde je pominjanje hrane prošireno barem jednom dodatnom rečju.

2.4. Obučavajući i test skup podataka

Inicijalni skup podataka se sastojao iz 10113 segmenata koje je bilo neophodno anotirati prema polaritetu sentimenta koji iskazuju. Kreiran je namenski program za anotiranje čiji je interfejs prikazan na slici 2. Anotatoru bi bio prikazan tekst segmenta i sentiment bi se označavao celobrojnom vrednošću 1 za pozitivan, 2 za negativan i 3 za neutralan sentiment. Izgled alata je predstavljen na slici 2.



Slika 2. *Interfejs alata za anotiranje*

Anotirani skup nije bio balansiran. Sadržao je 3466 segmenta sa pozitivnim, 1245 sa negativnim i 5224 sa neutralnim sentimentom. Odlučeno je da se iskoriste svi segmenti sa negativnim sentimentom i da se iz preostala dva skupa nasumice izabere po 1245 segmenata. Prema tome, kreiran je balansiran skup podataka u kom su se nalazila ukupno 3735 segmenta. Svaki segment je prvo pretprocesiran i potom transformisan u vektor reči kako bi bilo moguće primeniti algoritme mašinskog učenja za klasifikaciju.

2.5. Pretprocesiranje segmenata

U segmentima su se često javljali šablonski delovi koji nikada ne nose sentiment, poput nabrajanja koliko je kog jela poručeno i po kojoj ceni („2x Kapricoza 32cm“). Ovaj šablon se mogao predstaviti regularnim izrazom i delovi teksta koji su se slagali sa njim su zamenjeni specijalnom oznakom. Postoje i šabloni za koji je polaritet veoma teško odrediti, poput ocenjivanja jela po nestandardizovanoj skali („5/5“ ili „2/10“) koji su takođe zamenjeni. Dobar indikator sentimenta predstavlja polaritet emotikona koji se javlja uz pominjanje hrane. Formirana su dva regularna izraza za zamenu emotikona, jedan za pozitivne i jedan za negativne. Primećeno je da su se u segmentima često javljale reči napisane sa višestrukim ponavljanjem nekog samoglasnika kako bi se dočarao intenzitet iskazanog mišljenja.
Višestruka ponavljanja su ostala zabeležena, ali se broj ponavljanja tog karaktera redukovao na dva. Isto tako, srpski jezik je veoma morfološki složen jezik. Morfološka normalizacija je uključena kako bi se različiti oblici iste reči sveli na istu osnovu.
Primenjene su dve tehnike normalizacije: steming i lemantizacija. Korišćene su Python implementacije stemera [9] i lemantizera iz ReLDI biblioteke. Morfološka normalizacija je imala veliki doprinos u redukciji dimenzionalnosti problema. Inicijalni skup atributa na kom nisu primenjene tehnike pretprocesiranja brojao je 10358 atributa. Primenom lemantizacije skup atributa je redukovan na 8832 atributa, a steminga na 8609. Svaki segment je predstavljen kao vektor atributa (eng. *feature vector*). Cilj je predstaviti svaku normalizovanu reč ulaznog segmenta određenom brojčanom vrednošću, kako bi se formirani vektor mogao koristiti kao ulaz ili izlaz modela mašinskog učenja. Korišćene su tri šeme formiranja vektora atributa : binarna vektorizacija, frekvencija terma i tf-idf.

3. EKSPERIMENTALNA EVALUACIJA I REZULTATI

Eksperimentalna evaluacija se vrši radi određivanje performansi korišćenih algoritama mašinskog učenja za klasifikaciju teksta prema iskazanom sentimentu. Metrike performansi algoritama korišćene u ovom radu su tačnost, preciznost, odziv i F_1-mera.

Pojedini parametri svakog od četiri korišćena modela mašinskog učenja su optimizovani. Optimizacija je izvršena kros-validiranom grid-pretragom (eng. *cross-validated grid-search*) po koordinatnoj mreži parametara koji se optimizuju. Upotrebljena je implementacija GridSearchCV [11] iz Python biblioteke Scikit-learn. Vršena je kros validacija iz *k* delova (eng. *k-fold cross-validation*). Zasebno su optimizovani parametri za svaku kombinaciju izbora šeme kodiranja atributa, šeme selekcije atributa i tehnike normalizacije i te vrednosti su korišćene u daljoj evaluaciji.

Za Naive Bayes [12] klasifikator je vršena optimizacija parametra *alpha (α)* i predstavlja korektivni faktor. Najbolji rezultat tečnosti, preciznosti, odziva i F1 mere je postignut za slučaj gde je korišćen steming, tf-idf šema kodiranja ulaza i unigrami kao šema selekcije atributa. Postignuti su rezultati od 0,7831, 0,7891, 0,7831 i 0,7816 respektivno za α = 0,6. Za SVM vršena je optimizacija vrednosti parametra *C*. On predstavlja parametar penalizacije greške klasifikacije. Generalno najbolje rezultate klasifikacije je imao SVM [13] klasifikator. Najbolji rezultat tečnosti, preciznosti, odziva i F1 mere je postignut za slučaj gde je korišćena lemantizacija, frekvencija terma kao šema kodiranja ulaza i unigrami kao šema selekcije atributa.
Postignuti su rezultati od 0,8046, 0,8122, 0,8046 i 0,8059 respektivno za *C* = 0,6. Za Random Forest [14] klasifikator optimizovani su parametri broja stabala (*n_estimators*) i procenat atributa originalnog skupa podataka će preuzeti svaki estimator (*max_features*). Random Forest je pokazao najlošije rezultate klasifikacije od svih klasifikatora. Najbolji rezultat tečnosti, preciznosti, odziva i F1 mere je postignut za slučaj gde je korišćen steming, frekvencija terma kao šema kodiranja ulaza i unigrami kao šema selekcije atributa. Postignuti su rezultati od 0,7617, 0,7901, 0,7617 i 0,7614 respektivno za *n_estimators* = 70 i *max_features* = 0,9. Iako se logistička regresija [15] retko koristi za klasifikaciju teksta, u ovom slučaju su dobijeni veoma zadovoljavajući rezultati. Performanse mogu da pariraju onima koji su postignuti korišćenjem SVM-a. Za klasifikator zasnovan na logističkoj regresiji vršena optimizacija parametara *C* i *solver*. Najbolji rezultati su postignuti upotrebom lemantizacije, tf-idf šeme kodiranja ulaza i unigrama kao šeme selekcije atributa. Postignuti su rezultati od 0,8005, 0,8056, 0,8005 i 0,8021 respektivno za *solver* = *lbfgs* i *C* = 1,0.

4. ZAKLJUČAK

U ovom radu predstavljeno je jedno rešenje za izdvajanje delova teksta iz korisničkih recenzija koji predstavljaju nosilac sentimenta za određeno jelo i klasifikacija tih delova teksta prema osećanju koje je u njima iskazano na pozitivno, negativno ili neutralno korišćenjem metoda mašinskog učenja. Skup podataka se sastojao iz recenzija restorana pisanih na srpskom jeziku. Izdvojeno je 10000 recenzija u kojima su manuelno anotirana sva pominjanja hrane. Sve recenzije su tokenizovane i POS-tagovane. Izvršena je lemantizacija i parsiranje zavisnosti. Segmenti recenzija koji predstavljaju nosilac sentimenta za određeno jelo izdvojeni su na osnovu leksičkih veza i skupa pravila koje je svaka veza treba da zadovolji.
Test i trening skup podataka su formirani iz segmenata u kojima su pominjanja hrame proširena barem jednim tokenom. Sentiment svakog segmenta je manuelno anotiran. Tehnike pretprocesiranja korišćene u ovom radu su zamene kvantitativnih i kvalitativnih numeričkih vrednosti i emotikona. Pored lemantizacije, razmatran je i steming kao tehnika morfološke normalizacije teksta.
Kao šema kodiranja atributa korišćeni su binarna, frekvencija terma i tf-idf. Razmatrane šeme selekcije atributa su unigrami i unigrami kojima su pridruženi bigrami. Za klasifikaciju su korišćena ukupno četiri algoritma nadgledanog mašinskog učenja: Naive Bayes, SVM, Random Forest i logistička regresija. Rad bi mogao da se proširi u dva aspekta. Prvi se tiče uklanjanja šuma i poboljšavanje tehnika pretprocesiranja, poput ponalaženja i zamene pogrešno napisanih reči validnim oblikom iste i dodavanja rečnika sinonima.
Drugi aspekt tiče se detaljnije optimizacije parametara klasifikatora u cilju postizanja boljih performansi. Generalno, rezultate je moguće unaprediti proširivanjem skupa podataka anotiranjem više recenzija.

5. LITERATURA

[1] www.donesi.com
[2] Jason Huggins, *et al*, 2004. Selenium, https://www.seleniumhq.org
[3] Leonard Richardson 2014, BeautifulSoup4 https://www.crummy. com/software/BeautifulSoup
[4] Ljubesic, Nikola, Tomaz Erjavec and Darja Fiser. "Corpus-Based Diacritic Restoration for South Slavic Languages." *LREC* (2016).
[5] Ljubesic, Nikola and Tomaz Erjavec. "Corpus vs. Lexicon Supervision in Morphosyntactic Tagging: the Case of Slovene." *LREC* (2016).
[6] Ljubesic, Nikola, Filip Klubicka, Zeljko Agic and Ivo-Pavao Jazbec. "New Inflectional Lexicons and Training Corpora for Improved Morphosyntactic Annotation of Croatian and Serbian." *LREC* (2016).
[7] Agic, Zeljko and Nikola Ljubesic. "Universal Dependencies for Croatian (that work for Serbian, too)." *BSNLP@RANLP* (2015).
[8] Fišer, D., Ljubešić, N. & Erjavec, T. Lang Resources & Evaluation (2018). https://doi.org/10.1007/s10579-018-9425-z
[9] Milosevic, Nikola "Stemmer for Serbian Language. " *CoRR* abs/ 1209.4471 (2012): n. pag.
[10] Universal Dependencies, https://universaldependencies.org/#language-u
[11] GridSearchCV, Scikit-learn https://scikit-learn.org/stable/modules/generated/-sklearn.model_selection.Grid-SearchCV.html
[12] Leung K. M. (2007). "Naive Bayesian Classifier", Polytechnic University, Department of Computer Science, Finance and Risk Engineering.
[13] Support Vector Machines (SVM), Statsoft, http://www.statsoft.com/Textbook/Support-Vector-Machines
[14] Breiman L. (2001) "Random Forests, Machine Learning", Vol. 45. Issue 1, pp. 5-32.
[15] Logistic Regression Scikit-learn, https://scikit-learn.org/stable/modules/generated/-sklearn.linear_model.Logistic-Regression.html

Kratka biografija:

Marija Joksimović rođena je u Sremskoj Mitrovici 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Računarstva i automatike – Sistemi za istraživanje i analizu podataka odbranila je 2019.god. kontakt: marijajoksimovic@hotmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.896
DOI: https://doi.org/10.24867/07BE10Becejac

PREDIKCIJA CENE PROIZVODA NA OSNOVU OPISA NJEGOVIH KARAKTERISTIKA

PRODUCT PRICE PREDICTION BASED ON DESCRIPTION OF ITS CHARACTERISTICS

Milana Bečejac, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Tema ovog rada je rešavanje problema predikcije cene proizvoda na osnovu opisa njegovih karakteristika. Specificirani su, implementirani i verifikovani modeli, koji rešavaju ovaj problem. Namena ovih modela je pomoć kupcima u proceni da li je cena proizvoda adekvatna, kao i pomoć prodavcima u određivanju prikladne cene za svoj proizvod.*

Ključne reči: *Predikcija cene, Neuronske mreže, Karakteristike proizvoda, Word2Vector, BagOfWords, LSTM*

Abstract – *The goal of this paper is solving the problem of prediction of price based on products' characteristics. It describes the specification, implementation, and verification of such a system. The purpose of this system is to help buyers find a product with a fair price and to assist sellers in determining a suitable price for their products.*

Keywords: *Price Prediction Neural networks, Product characteristics, Word2Vector, BagOfWords, LSTM*

1. UVOD

U današnje vreme može biti veoma teško odrediti da li neki proizvod na tržištu stvarno vredi toliko kolika mu je cena i detalji mogu dovesti do velike razlike. Na primer, na cenu odeće mogu uticati sezona prodaje ili brend, dok na elektroniku najviše utiče specifikacija proizvoda [1]. Prodavcima je teško da odrede cenu svom proizvodu i da ta cena bude odgovarajuća kupcima. Da bi odredili cenu, prodavci mogu istraživati tržište i tražiti slične proizvode ili pitati druge trgovce za sugestije. Međutim, ove metode oduzimaju puno truda i vremena i mogu dovesti do troškova koji su veći od same vrednosti proizvoda [1].

S druge strane, zbog razvoja tehnologije, sve više je zastupljena kupovina preko interneta i iz tog razloga mnoge *online* prodavnice istražuju načine da kupcima predlože svoje proizvode kroz svoje sajtove koristeći sisteme za preporuku proizvoda [2]. Takođe, pretragom sajtova kupci mogu i samostalno da uporede cene različitih proizvođača za isti proizvod i na taj način odluče gde da izvrše kupovinu.

Ovaj način odlučivanja o kupovini može biti neefikasan ako kupac ne uloži dovoljno vremena u traženju najadekvatnije ponude. Uvođenje adekvatnog metoda za procenu cene proizvoda bi sa jedne strane pomoglo potencijalnim kupcima tako što bi im potvrdio da je zahtevana cena prikladna, dok bi prodavcima pomogla pri odabiru odgovarajuće cene.

Na sajtu *kaggle.com* [3] postoji takmičenje pod nazivom "Mercari Price Suggestion Challenge" i u ovom radu će biti predstavljeno više modela za rešavanje problema predikcije cene proizvoda na osnovu opisa njegovih karakteristika: opisa proizvoda u tekstualnoj formi, naziva brenda, stanja i kategorije proizvoda. Sva rešenja za objektivno određivanje cene proizvoda u ovom radu su realizovana kao modeli za predviđanje cene obučeni na osnovu podataka preuzetog sa *kaggle.com* sajta [3]. Pri implementaciji modela, velika pažnja je posvećena analizi i obradi skupa podataka. Za pretprocesiranje korišćene su različite tehnike obrade teksta kako bi se dobili što adekvatniji rezultati.

Na Kaggle-ovom takmičenju zahtevano je korišćenje RMSLE i iz tog razloga, ovaj metod je korićen kao glavna metrika evaluacije rezultata sitema opisanog u ovom radu. Skup podataka je podeljen na 70% trening, 20% test i 10% validacioni skup podataka.

2. METODOLOGIJA

Sva programska rešenja implementirana za potrebe rada su izrađena u programskom jeziku *Python*.
Za potrebe ovog rada je obučeno 14 modela nad skupovima podataka različitih veličina u cilju utvrđivanja balansa između ostvarenih performansi i vremena utrošenog na treniranje. Pre pokretanja svakog modela izvršeno je predprocesiranje skupa podataka

2.1. Predprocesiranje skupa podataka

Nakon analize podataka, odlučeno je da nad skupom podataka, koji će biti korišćen za treniranje modela bude izvršeno pretprocesiranje kako bi se dobili što bolji rezultati predikcije cene proizvoda.
Metode obrade teksta koji su izvršeni nad obeležjima naziv proizvoda i opis proizvoda kao što je predloženo u radu [4]:

- Pretvaranje svih slova reči iz velikih u mala slova
- Uklanjanje znakova interpukcije
- Uklanjanje stop-reči reči koje se smatraju nebitnim u engleskom jeziku, kao što su: and, the, for, a, in, to,...
- Pretvaranje svih reči u koren (*stem*) te reči
- Dopuna nedostajućih vrednosti, zamena sa stringom "missing". Dopuna je izvršena nad tekstualnim obeležjima opis, brend i kategorija proizvoda.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bila dr Jelena Slivka, docent.

Nad obeležjima brend i kategorija proizvoda izvršene su sve prethodne metode pretprocesiranja, kao i pretvaranje u numeričke vrednosti *SKLearn One-hot* encoding metodom.

2.2. Inicijalni RNN model

Inicijalni model je bila rekurentna neuronska mreža, koja je kao ulaz primala skup podataka, nad kojim je izvršeno samo pretvaranje kategoričkih obeležja brenda i kategorije proizvoda u numeričke vrednosti tih obeležja, kao što je opisano u prethodnom poglavlju. Rekurentna neuronska mreža (RNN) je tip veštačke neuronske mreže koja se obično koristi u obradi prirodnog jezika. RNN je dizajnirana tako da prepoznaje sekvencijalne karakteristike podataka i da koristi paterne za predviđanje najverovatnijeg sledećeg scenarija [5]. Naš inicijalni model je imao *Embedding* slojeve, po jedan za opis, naziv, brend, kategoriju i stanje proizvoda, 2 GRU (*Gated Recurrent Unit*) sloja, a za aktivaciju je korišćena linearna funkcija. GRU je varijanta LSTM-a, koja takođe rešava problem nestajućeg gradijenta, ali je jednostavnija i brža od LSTM-a (*Long Short Term Memory*) [6]. Ova arhitektura je dizajnirana po uzoru na rad [7]. Kao mera za testiranje ovog modela korišćen je RMSLE.

2.3. *Random Forest* algoritam

U radu je isproban i *Random Forest* algoritam, nad skupovima podataka različitih veličina i sa različitim vrednostima parametara. Performanse ovog algoritma testirane su pomoću mere tačnosti MAE (*Mean Absolute Error*). U radu [8] je predložena ova mera performansi za *Random Forest* algoritam i, nakon analize rezultata, utvrđeno je da je tačno tvrđenje iz rada [8] i da *Random Forest* daje loše rezultate pri rešavanju predikcije cena proizvoda. Iz tog razloga odlučeno je da se pažnja posveti rešavanju problema korišćenjem modela neuronske mreže, analizi i pretprocesiranju skupa podataka.

2.4. *BagOfWords* i *Word2Vector*

Većina modela opisanih u ovom radu vrši predikciju samo na osnovu opisa proizvoda u tekstualnom obliku, ali su isprobani i modeli koji u obzir uzimaju i stanje, status, kategoriju i brend, kako bi se utvrdilo da li se dodavanjem ostalih obeležja može poboljšati rezultat.

U narednim modelima, opisi svakog proizvoda iskazani su kao vreću reči (eng. *Bag of words*) ili vektore reči (eng. *Word2Vector*) i njihova reprezentacija predstavljala je ulaze u neuronsku mrežu.

Sledeće unapređenje modela bilo je to da *Word2Vector* reprezentacija opisa proizvoda predstavlja ulaz za LSTM sloj neuronske mreže. Modeli sa *softmax* aktivacionom funkcijom sortiraju proizvode u 12 ili 20 različitih kateogrija cena, kao što je urađeno i u radu [9], dok modeli sa linearnom aktivacionom funkcijom pokušavaju da pogode tačnu cenu za svaki testni proizvod. Kod modela sa *softmax* aktivaciom, kategorije su napravljene tako da su relativno balansirane. Za evaluaciju ovih modela korišćena je tačnost (*Accuracy*) kada je u pitanju *softmax* aktivacija, a *Percent error* i RMSLE kada je u pitanju linearna aktivaciona funkcija.

Nakon pretprocesiranja skupa podataka opisanog u prethodnom poglavlju, izvršeno je enkodiranje obeležja, tako da budu pogodni za slanje na ulazni sloj novih modela. U početku je najveći fokus bio na opisu proizvoda, jer je bila pretpostavka da će on najviše uticati na rezultate predikcije. Opis svakog proizvoda predstavljen je na dva različita načina.

Za implementaciju "vreće reči" napravljen je vektor, veličine rečnika svih različitih reči koje se pojavljuju u koloni opis proizvoda iz skupa podataka. Zatim je svaki opis predstavljen kao vektor veličine jednakoj broju svih različitih reči koje se pojavljuju u svim opisima. Za svaki vektor je zabeležemo prisustvo te reči u odgovarajućem opisu, 1 ako je reč prisutna, 0 ako nije.

2.5. Jednostavna troslojna neuronska mreža

Model 1 je jednostavna troslojna neuronska mreža kojoj je kao ulaz prosleđena prethodno opisana vreća reči. Ovaj model na izlazu ima *Softmax* aktivacionu funkciju i zbog toga mera je za evaluaciju ovog modela tačnost modela (*Accuracy*). Cilj modela je da svaki proizvod svrsta u jednu od 12 kategorija cena. Ovaj model je isproban na malom trening skupu podataka od 2.000 primera i sa malim brojem epoha i dobijena je tačnost od 23,7% nad trening skupom i 16,5% nad test skupom. Ovakvi loši rezultati su i bili očekivani jer se treniranje vršilo sa malim brojem primera, što u startu nije dobro kada se koriste neuronske mreže. Takođe, trebalo je jako puno vremena za njegovo treniranje zbog veoma dugačkih vektora koji su predstavljali ulaze, pa je odlučeno da se koristi *Word2Vector* enkodiranje opisa proizvoda, jer proučavajući literaturu, posebno rad [9] ustanovljeno je da se na taj način mogu poboljšati performanse.

U modelu 2, korišćena je neuronska mreža sa istom arhitekturom kao i u modelu 1, razlika je što je ulazni sloj predstavljao vektore enkodiranih vrednosti opisa proizvoda, koje su dobijene korišćenjem GloVe fajla (glove.6b.100d), kako je predloženo u radu [9]. GloVe fajl sadrži 100 dimenzionalne vektore pretreniranih vrednosti za reči engleskog jezika dobijenih na osnovu učestalosti pojavljivanja te reči. Za svaki proizvod formiran je vektor vrednosti, gde svaka vrednost odgovara jednoj reči iz opisa proizvoda i predstavlja prosek vrednosti za tu reč iz GloVe fajla. Model je bilo moguće trenirati i sa većim skupom podataka, a da pritom ne dođe do prevelikog gubitka vremena u treniranju. Najbolji rezultat je dobijen nad skupom od 50.000 primera i iznosi 19,18% tačnosti nad trening skupom i 16,46% tačnosti nad validacionim skupom.

2.6. Neuronska mreža sa LSTM slojem

Nakon toga, odlučeno je da se sačuva redosled reči u opisu proizvoda, kako bi se i to uzelo u obzir prilikom predikcije cene. Da bi ovo bilo postignuto, jednostavna neuronska mreža zamenjena je neuronskom mrežom sa LSTM slojem, na koji se šalju pretrenirane, enkodirane *Word2Vector* vrednosti opisa proizvoda kao ulazi. Ovaj model je odabran proučavajući literaturu [5], gde se navodi da je pogodan za rešavanje ovakve vrste problema predikcije, a da takođe rešava i problem nestajućeg Gradienta[1] (sprečavanje napredovanja procesa obučavanja neuronske mreže tako što gradijenti prilikom treniranja dobijaju previše malu vrednost). Ovaj model se bazira na

[1] https://ayearofai.com/rohan-4-the-vanishing-gradient-problem-ec68f76ffb9b , preuzeto 08.09.2019.

tome da se opisi proizvoda šalju na LSTM sloj gde je *Word2Vector* enkodirana vrednost za svaku reč korišćena za svaki korak u LSTM. Ovaj proces se ponavlja za prvih X reči svakog opisa. Utvrđeno je da je tačna tvrdnja u radu [9] i da se dobija približna tačnost modela i kada se proces ponavalja za maksimalnih 415 reči i kada se ponavlja za 72 reči, a dolazi se do znatne uštede vremena. Iz tog razloga, odlučeno je da vrednost ovog parametra bude 72. Izlaz poslednjeg LSTM koraka ide na *Softmax* izlazni sloj i izlaz je vektor dužine 12, gde svaki izlaz odgovara jednoj grupi cena.

Model 4 ima istu arhitekturu kao i model 3, samo su umesto pretreniranih *Word2Vector* vrednosti, korišćene *Word2Vector* enkodirane vrednosti trenirane nad našim skupom podataka.

Od reči iz opisa proizvoda formiraju se vektori enkodiranih vrednosti reči, korišćenjem fajla kreiranog po uzoru na GloVe fajl, ali se u ovom fajlu za svaku različitu reč iz svih opisa proizvoda kreira vektor vrednosti u zavisnosti od njene učestalosti pojavljivanja u svim opisima. Svaka vrednost dobijenog vektora odgovara jednoj reči iz datog opisa i predstavlja prosek vrednosti za tu reč iz kreiranog fajla. Ovako dobijeni vektori predstavljaju ulaze za LSTM neuronske mreže.

2.7. Neuronska mreža sa LSTM slojem i potpuno povezanim slojem

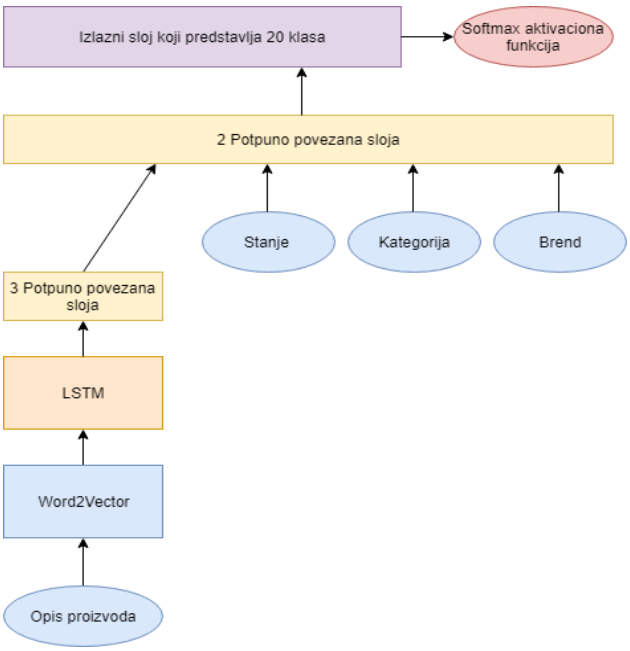
Zatim je odlučeno na osnovu pročavanje rada [4] da se prilikom predikcije cene proizvoda, osim njegovog opisa, uzmu u obzir i kategorija, naziv brenda i stanje kako bi se poboljšale performance modela. Kod modela 6 opisi proizvoda se šalju na LSTM sloj na isti način kao i kod modela 3 i 4, samo što izlaz poslednjeg LSTM koraka, zajedno sa One-hot enkodiranim vrednostima za brend, kategoriju i stanje proizvoda predstavlja ulaz u potpuno povezan sloj, koji ima *Softmax* aktivacionu funkciju i predstavlja vektor dužine 20, gde svaki izlaz odgovara jednoj grupi cena. Broj klasa je povećan kako bi se poboljšala uspešnost našeg modela.

Model 7 je potpuno iste arhitekture kao i model 6, samo je aktivaciona funkcija linearna.

2.8. Neuronska mreža sa LSTM slojem i 3 potpuno povezana sloja

Naredno unapređenje modela predstavlja zamenu postojećeg potpuno povezano sloja sa 3 potpuno povezana sloja, kako bi se poboljšali rezultati predikcije. Testirali smo model 8 i 10 sa *Softmax* aktivacionom funkcijom, gde kod modela 10 postoje 12 klasa, a kod modela 8 postoji 20 klasa, koje predstavljaju rezultat predikcije. Model 9 je testiran sa linearnom aktivacionom funkcijom. Modeli koji vrše klasifikaciju su se pokazali kao najbolji od svih isprobanih modela sa *Softmax* aktivacijom, a takođe je model 9 imao RMSLE koji je najbolji u istraživanju i spao bi u polovinu najboljih rezultata ostvarenih na *Kaggle*-ovom takmičenju Arhitektura jednog od modela sa najboljim rezultatima je prikazana na slici 1.

Modeli 11 (*Softmax* aktivaciona funkcija, 12 klasa), 12 (*Softmax* aktivaciona funkcija, 20 klasa) i 13 (linearna aktivaciona funkcija), imaju arhitekturu kao i prethodno navedeni modeli, samo što se pri predviđanju ne uzimaju u obzir naziv brenda, kategorija i stanje proizvoda, već samo opis proizvoda. Testiranjem, došlo se do zaključka da ova 3 modela daju lošije rezultate od prethodnih.



Slika 1. Arhitektura modela sa najboljim performansama

3. EVALUACIJA I REZULTATI

Za glavnu metriku evaluacije uzet je RMSLE (*Root Mean Square Error)* (slika 2), jer daje veću kaznu za podcenjivanje cene, nego za veću procenu cene. Korišćenje ovog metoda evaluacije zahtevano je i na *Kaggle*-ovom takmičenju i korišćeno u radovima [1][9], pa je odlučeno da se i za verifikaciju modela opisanih u ovom radu koristi ovaj metod evaluacije za modele koje imaju linearnu aktivacionu funkciju, dok je za evaluaciju neuronske mreže koja ima softmax aktivacionu funkciju korišćena tačnost (*Accuracy*).

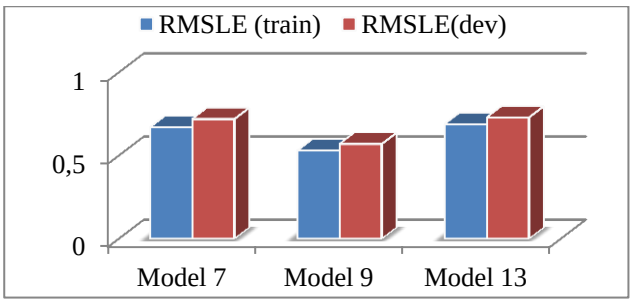
$$\varepsilon = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(p_i + 1) - \log(a_i - 1))^2}$$

- ε – vrednost RMSLE
- n – ukupan broj observacija u javnom/privatnom skupu podataka
- p_i – predikcija cene proizvoda
- a_i – stvarna cena proizvoda
- $\log(x)$ – prirodni logaritam od x

U tabeli 3 su prikazani najbolji rezultati predikcije modela sa linearnom aktivacionom funkcijom, dok je na slici 4 prikazan dijagram ovih rezultata.

Tabela 3. *Najbolji rezultati predikcije modela sa linearnom aktivacionom funkcijom*

	RMSLE (train)	RMSLE (dev)
Model 7	0,67	0,72
Model 9	0,53	0,57
Model 13	0,69	0,73



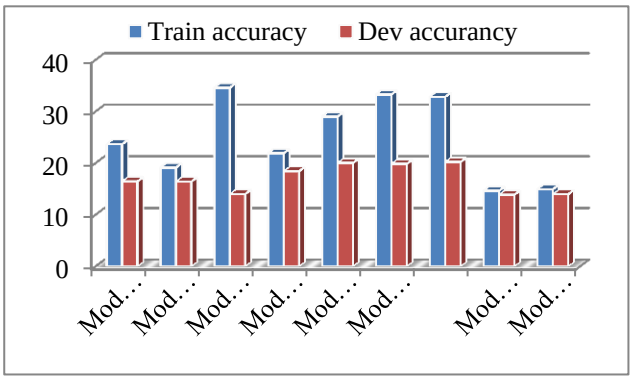
Slika 4. *RMSLE za modele sa lineanom aktivacionom funkcijom*

U tabeli 5 su prikazani najbolji rezultati predikcije modela sa *softmax* aktivacionom funkcijom, dok je na slici 4 prikazan dijagram ovih rezultata.

Tabela 4. *Najbolji rezultati predikcije modela sa softmax aktivacionom funkcijom*

	Train accuracy	**Dev accuracy**
Model 1	23,7	16,5
Model 2	19,18	16,46
Model 3	34,65	14,04
Model 4	21,94	18,46
Model 6	29,04	20,06
Model 8	33,32	19,88
Model 10	32,86	20,26
Model 11	14,67	13,9
Model 12	14,97	14

Na osnovu ovih dijagrama dolazimo do zaključka da najbolje rezultate daju modeli 9 kada je u pitanju linearna aktivaciona funkcija sa ostvarenim RMSLE = 0.53 nad trening, odnosno RMSLE = 0.57 nad validacionim skupom podataka, što bi spadalo u prvu polovinu najboljih ostvarenih rezultata na Kaggle-ovom takmičenju, odnosno, model 10 kada je u pitanju *softmax* aktivaciona funkcija sa ostvarenom tačnošću od 32,86% nad trening skupom i 20.26% nad validacionim skupom podataka.



Slika 5. *RMSLE za modele sa lineanom aktivacionom funkcijom*

4. ZAKLJUČAK

U ovom radu su prikazani modeli za utvrđivanje adekvatne cene proizvoda na osnovu njegovih karakteristika, kao što su opis, naziv brenda, kategorija i stanje proizvoda. Skup podataka korišćen za treniranje svih modela je javno dostupan na sajtu *kaggle.com* [3].

Skup podataka je podeljen na 70% trening, 20% test i 10% validacionih podataka. Isprobani su modeli neuronske mreže sa različitim arhitekturama i sa različitim parametrima. Obučeno je 14 modela nad skupovima podataka različitih veličina u cilju utvrđivanja balansa između performansi i vremena treniranja.

Kao rešenje sa najboljim rezultatima je neuronska mreža sa LSTM slojem, čiji su ulazi enkodirane *Word2Vector* vrednosti trenirane sa našim skupom podataka. Izlaz ovog sloja ujedno predstavlja i ulaz u 3 potpuno povezana sloja mreže. Izlaz ovog dela mreže zajedno sa *One-hot encoding* reprezentacijama stanja, naziva brenda i kategorije proizvoda predstavlja ulaz u nova 3 potpuno povezana sloja sa linearnom aktivacionom funkcijom. RMSLE dobijen evaluacijom ovog modela iznosi 0.5732.

U budućnosti, modeli se mogu unaprediti analizom proizvoda čija procenjena cena se drastično razlikuje od stvarne. Takođe, jedno od unapređenja može biti analiza opisa proizvoda, kako bi se utvrdili delovi opisa koji najviše utiču na cenu, čime bi se smanjila veličina ulaza u neuronsku mrežu.

Rešenja problema predikcije cene proizvoda na osnovu njegovih karakteristika, koja su tema ovog rada, mogu pomoći kako prodavcima u određivanju cene proizvoda na tržištu, tako i kupcima u pronalaženju proizvoda po najadekvatnijoj ceni.

5. LITERATURA

[1] Su, Beichen. An Application of Recurrent Neural Network: Prediction of Items' Price by Description. Diss. UCLA, 2018.
[2] Kim, Kyoung-jae, and Hyunchul Ahn. "A recommender system using GA K-means clustering in an online shopping market." Expert systems with applications 34.2 (2008): 1200-1209.
[3] https://www.kaggle.com (pristupljeno u septembru 2019.)
[4] https://github.com/ChenglongChen/tensorflow-XNN/blob/master/doc/Mercari_Price_Suggesion_Competition_ChenglongChen_4th_Place.pdf (april 2019.)
[5] https://searchenterpriseai.techtarget.com/definition/recurrent-neural-networks (septembar 2019.)
[6] https://medium.com/mlrecipies/deep-learning-basics-gated-recurrent-unit-gru-1d8e9fae7280 (septembar 2019.)
[7] https://github.com/Sebastianvarv/MercariPriceSuggestion (maj 2019.)
[8] https://github.com/gdayal80/mercari/tree/local/Mercari%20Price%20Suggestion%20Challenge (maj 2019.)
[9] Raygada, Javier. "Product Price Suggestions for Online Marketplaces."

Kratka biografija:

Milana Bečejac rođena je u Zrenjaninu 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnika i računarstvo odbranila je 2019. god.
kontakt: milana.becejac@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE11Jovicic

RAZVOJ VEB APLIKACIJE TRIO MJUZIK

DEVELOPMENT OF A WEB APPLICATION TRIO MUSIX

Vladimir Jovičić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – *Zadatak rada predstavlja analizu veb aplikacije koja korisnicima pruža usluge trgovine preko interneta muzičkih instrumenata i opreme, zakazivanje i gledanje kurseva kao i servis instrumenata. Biće posmatrani aspekti programera i korisnika, kako implementacije tako i dizajn za krajnjeg korisnika kako bi on brzo pronašao ono što veb stranica nudi.*

Ključne reči: *Prodavnica, servis, kursevi*

Abstract – *The task of the paper is to analyze a web application that provides users with online music instruments and equipment trading services, to schedule and watch courses, and to service instruments. Aspects of developers and users will be looked at, both implementation and design for the end user to quickly find what the website has to offer.*

Keywords: *Shop,, service, courses*

1. UVOD

U današnje vreme, *web* tehnologije i *web* programiranje su zastupljeni u velikoj meri. Moderne *web* aplikacije se najčešće kreiraju tako što se cela aplikacija razdvoji u dva dela : klijentska strana i serverska [1] strana. Klijentska strana predstavlja ono što korisnik vidi i čime upravlja, a serverska strana predstavlja ono što se dešava "u pozadini" tj. obrada zahteva koju korisnik pošalje. Klijentka strana se najčešće naziva i *frontend*, a serverska strana se najčešće naziva i *backend*. Serverska strana uglavnom upravlja sa bazom podataka, izvršava određene operacije nad istom, i šalje klijentskoj strani povratnu informaciju o izvršenim operacijama nad bazom.

Da bi se to olakšalo i da bi se omogudila dinamička web stranica koja bi reflektovala korisničke unose uveden je *CGI* (*Common Gateway Interface*). To je standard koji je uveden za interfejsing spoljnih aplikacija sa web serverima. *CGI* je imao tu manu što je svaki zahtev morao biti pokrenut kao poseban proces.

Velika većina programskih jezika pružaju gotove biblioteke za razne zadatke, ali isto tako, vedina zahteva uključivanje dodatnih biblioteka za rad sa web aplikacijama i npr. pisanje *HTML* koda.

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

Web aplikacije se danas sve više i više dele u po raznim kriterijumima u zavisnosti od njihove namene. Na osnovu toga, neke od vrsta *web* aplikacija mogu biti: *Online* kupovina/prodaja, društvene mreže, sajtovi za edukaciju i gledanje kurseva, sajtovi za zabavu, bankarstvo… Takođe je moguća i kombinacija bilo koje dve predhodno napomenutih vrsta. Na primer sajtovi za *online* kupovinu su usko povezani sa sajtovima za bankarstvo jer se plaćanje vrši uglavnom preko neke centralne banke.

2. OPIS REŠAVANOG PROBLEMA

Mnoge modernije i razvijenije *web* aplikacije se kombinuju često i sa više od 2 vrste pri čemu nastaju multi namenske *web* aplikacije. Sajtovi društvenih mreža pored toga što nude kontakt sa određenim "prijateljima", nude i dodatne forume za oglase, video *chat*, *entertainment*, pregled novosti (kako "prijatelja" tako i vesti u globalu). Time je sajt obezbedio sebi širok spetkar korisnika što znači da što više ljudi poseduje sajt na dnevnoj bazi. Ako je sajt napravljen da bude "*user friendly*", što podrazumeva da je jednostavan za upotrebu svim posetiocima, čak i ljudima koji se malo manje razumeju u tehnologiju, tada je sajt sebi osigurao visoku posedenost na dnevnom nivou.

2.1. *Web* aplikacije Trio „Trio Music"

Web sajt za *online* prodaju muzičkih instrumenata sadrži u sebi (kao i svaka *online* prodavnica) izbor svih artikala koje korisnik može da kupi. Uz svaki artikal uglavnom se prikazuje i slika, cena, marka (ako je npr. u pitanju gitara onda se navodi i koja je debljina žica koje dolaze uz nju, boja, vrsta drveta od koje je napravljena i slično). Pretraga kod ovakvih sajtova mora da bude veoma detaljna ali i jednostavna.

Svaki instrument (kao i sve ostalo) je sporno kvarovima. Ponekad se kvar može otkloniti i to znatno jeftinije košta nego da se kupuje nov artikal. Za neke sitnije kvarove poželjno je imati kratke opise koje bi korisnik sam mogao da primeni na svom kvaru kako bi ga brzo i efikasno uklonio ali ta opcija je veoma rizična jer iz neiskustva može slediti nanošenje još većeg kvara.

Postoje i ljudi koji žele da nauče da sviraju a ne znaju kako početi, ili oni koji znaju svirati a žele da nauče još neke dodatne tehnike kako bi se usavršili. Nekima je lakše da gledaju lekcije direktno sa svojih uređaja na sajtu, a nekima je lakše interaktivno sa mentorom da nauče nešto novo pri čemu sajt nudi zakazivanje kurseva (individualno ili grupno) jer možda smatra da de svoje veštine savladati na taj način brže i/ili bolje.

3. OPIS KORIŠĆENIH TEHNOLOGIJA

Sa developerske strane gledano postoji više načina da se implementira sajt za *online* kupovinu/servis i kurseve sviranja muzičkih instrumenata. Treba voditi računa o nekim osnovnim pravilima kako bi se olakšala ne samo implementacija i razvoj ved i samo održavanje *web* aplikacije što podrazumeva konstantno menjanje izgleda/sadržaja stranice, načina upotrebe i slično. Jer ako se sajt koristi frekventno uvek de se nadi zamerke i određeni *bug*-ovi koji se mogu (i trebaju) popraviti.

3.1 *Backend*

Zbog komplikacije ovakve aplikacije, neretko se dešava da više ljudi radi na njoj. Često su poslovi podeljeni na celine *stack*-ova. To znači da jedna ili više osoba rade *frontend*, jedna ili više osoba rade *backend [2]*, jedna ili više osoba se bavi bazom podataka. Takođe je mogude imati i testere koji glume aktivnog korisnika nad gotovim, ali još ne objavljenom aplikacijom, i tako traže mane koji bi se mogle popraviti pre same produkcije

3.1.1 *SOAP* princip

SOAP (*Simple Object Access Protocol*) [3] je komunikacioni protokol, nezavisan od platforme, baziran na *XML*-u koji se koristi za razmenu informacije između aplikacije preko *HTTP* protokola. Razvijen je kako bi omogućio jednostavnu komunikaciju tekstualnim sadržajem preko *HTTP* komunikacionog protokola koji je prilagođen upravo razmeni tekstualnih sadržaja. Protokol je nezavisan od programskog jezika, platforme i jednostavno proširiv.

SOAP protokol omogućuje komunikaciju između aplikacija koje rade na različitim operativnim sistemima i različitim tehnologijama. Jedini zahtev koji moraju zadovoljiti je da u komunikaciji mogu koristiti *HTTP* protokol. Aplikacije razmenjuju poruke dogovorenog formata.

3.1.2 *REST* princip

REST (*REpresentational State Transfer*) [4] arhitekturalni stil je razvijen od strane *Technical Architecture Group (TAG)* paralelno sa *HTTP* 1.1. Ovaj stil je dobio najveću primenu na *World Wide Webu*. *REST* arhitektura se tipično sastoji od klijenta i servera. Klijent inicira zahtev serveru, server procesira zahtev i vrada odgovor klijentu. Glavni ciljevi REST-a su skalabilnost, generalizovanost *interface*-a i nezavisan broj komponenti.

Korišdenje *HTTP* metoda podrazumeva da se pri upotrebi *REST*-a koriste standardni zahtevi *HTTP* metode, a to su: kreiranje i čuvanje (***C**reate*), učitavanje postojećeg (***R**ead*), menjanje postojećeg (***U**pdate*) i brisanje (***D**elete*). Ove 4 metode se zovu (na osnovu početnih slova na engleskom) i *CRUD* metode.

Prenos resursa se najčešće vrši preko *JSON* objekata što predstavlja jedan od lakših tekstualnih otvorenih standarda dizajniran za čitljivu razmenu podataka. Ekstenzija datoteke s podacima u *JSON*-ovom formatu je *.json*, dok je meta oznaka (*MIME* format) application/json. *JSON* u sebi može imati i druge *JSON* objekte.

```
{
    "ime":"Vladimir",
    "prezime":"Jovicic",
    "god":24,
    "struka":"Master inzenjer"
}
```

Slika 3.1.2.1 Prime *JSON* objekta

3.2 *Frontend*

Frontend [5] se bavi onim delom *web* aplikacije koji korisnik vidi. Ono što je prikazano u samom pretraživaču. On se može podeliti u 3 osnovna dela: Statički i dinamički sadržaj koji se radi uz pomod *HTML*-a (*hyper text makrup language*), upravljanje funkcionalnostima (*JavaScript* ili neka njegova biblioteka ili *framework*) i dizajniranje, opisivanje pozicija i pokreta sadržaja pomodu *CSS*-a (*Cascading Style Sheet*).

3.2.1 *HTML*

HTML (*hyper text makrup language*) [6] je opisni jezik specijalno namenjen opisu *web* stranica. Pomoću njega se jednostavno mogu odvojiti elementi kao što su naslovi, paragrafi, citati i slično. Pored toga, u *HTML* standard su ugrađeni elementi koji detaljnije opisuju sam dokument kao što su kratak opis dokumenta, ključne reči, podaci o autoru i slično. Ovi podaci su opštepoznati kao meta podaci i jasno su odvojeni od sadržaja dokumenta. Svi *HTML* dokumenti bi trebalo da počinju sa definicijom tipa dokumenta *DTD* (*Document Type Definition*) koji pregledaču definiše po kom standardu je dokument pisan.

3.2.2 *CSS*

CSS (*Cascading Style Sheets*) je jezik formatiranja pomoću kog se definiše izgled elemenata *web* stranice. Prvobitno, *HTML* je služio da definiše kompletan izgled, strukturu i sadržaj *web* stranice, ali je od verzije 4.0 *HTML*-a uveden *CSS* koji bi definisao konkretan izgled, dok je *HTML* ostao u funkciji definisanja strukture i sadržaja. *CSS* je u određenoj formi postojao još u začecima *SGML*-a 1970-ih godina. Kako je *HTML* postajao komplikovaniji, davao je sve više mogućnosti za definiciju izgleda elemenata, ali je istovremeno postajao nečitljiviji i teži za održavanje. Različiti pretraživači su prikazivali dokumente na različite načine, i postojala je potreba za doslednom tehnikom definisanja prikaza elemenata na stranici.

3.2.2 *JavaScript*

Javascript [8] je je dinamički, slabo tipiziran i interpretiran programski jezik visokog nivoa. Pored *HTML*-a i *CSS*-a, *JavaScript* je jedna od tri vodeće tehnologije za definisanje sadržaja na *webu*. Većina *web* sajtova koristi *Javascript* a svi moderni pretraživači ga podržavaju bez potrebe za instaliranjem dodataka. Kombinovan sa *HTML* jezikom i *CSS*-om *Javascript* čini *DHTML* (*Dynamic HTML*). *Javascript* je jezik zasnovan na prototipovima sa funkcijama prvog reda, što ga čini jezikom višestruke paradigme koji podržava objektno-orijentisani, imperativni i funkcionalni način programiranja.

4. OPIS REŠENJA PROBLEMA

Problem posmatran sam po sebi je pre svega izazovan sa developerske strane posmatrano. Potrebno je da se iskonfigurišu okruženja u kojima de se raditi kao i bazu podataka. Zbog svoje složene prirode i činjenice da de se sve razdvojiti na više pod-aplikacija, potrebno je svaku od njih posebno podesiti.

4.1 Glavna arhitektura

Glavna arhitektura aplikacije se sastoji iz 3 zasebna *frontend* dela i 3 zasebna *backend* dela i jednom zajedničkom bazom podataka. Glavna ideja je da svaki frontend deo "komunicira" sa svojim korespodentnim backend delom. Svaki *backend* deo je povezan sa istom bazom podataka što znači da svaki od njih ima zajedničku konfiguraciju pristupu bazi podataka. Time je omogudena podela resursa na sve delove aplikacije i ciljnim pozivanjem upita iz određenih delova dobijaju se određeni entiteti za daljnju obradu. U svakom od *backend* delova se nalazi određeni endpoint koji je potreban da se pozove kako bi se izvršila neka akcija sa obradom podataka (učitavanje iz baze, izvršavanje nekih proračuna…). To je ujedno i jedna od stvari koja konzolne aplikacije razlikuje od proceduralnih (konzolne se izvršavaju u jednom toku,

4.1.1 Baza podataka

Zbog potencijalne velike komplikacije i opširnosti baze podataka, preporučeno je raditi u nekoj od relacionih baza podataka poput *MySQL [9]*. On predstavlja višenitni (*multithread*) je višekorisnički *SQL* sistem za upravljanje bazama podataka. Sistem radi kao server, obezbeđujudi višekorisnički interface za pristup bazi podataka.

4.2 Interfejs

Interface same *web* aplikacije je prilagođen standardizaciji dizajna modernih *web* aplikacija radi održavanja konzistentnosti. Pri ulasku na sajt, na samoj početnoj strani aplikacije, korisnik ima mogućnost da izabere jednu od 3 ponuđene usluge koje pruža aplikacija. Izbori treba da budu jasno definisani i nedvosmisleni, sve na jednom mestu uredno raspoređeni ili jedna ispod druge ili jedna iznad druge. Pored svega toga ima jedna poruka koja dočekuje korisnika sa dobrodošlicom. Boje treba da budu blage jačine da korisnika „ne udara" u oči kako bi mu lakše pala koncentracija. Izborom bilo koje od 3 opcije korisnik se prosleđuje na određeni deo aplikacije koji je izabrao.

4.2.1 Zajednički interfejs

Da bi aplikacija bila konzistentna, potrebno je da sva 3 dela aplikacije podrže ista glavna svojstva da bi korisnik imao osećaj da je i dalje na istom sajtu. Neke od tih osobina su: Ista pozadina i boja pozadine mora da bude prisutna na svim delovima, isti font mora biti prisutan što podrazumeva istu veličinu, stil, boju i način pisanja, sva obaveštenja i sve poruke koje iskaču, kao i sve animacije moraju biti isto napravljene Pri svakom dolasku na neku od delova aplikacije, pojavljuje se dobrodošlica i u svakom trenutku treba da piše sekcije u kojoj se korisnik trenutno nalazi tako da on u svakom trenutku zna stanje aplikacije i ako se nalazi u onom delu u kome ne želi da bude, da lako može da pređe u sekciju aplikacije koja mu je potrebna.

4.2.2 Prodavnica

Sekciju prodavnice korisnik posećuje kada je odlučio da želi da iz širokog asortimana izabere neki proizvod, vidi recenziju, pogleda da li odgovara cena i zatim putem online kupovine ili lično da kupi i preuzme taj proizvod.

4.2.2.1 Početna stranica

Sama početna strana treba da ima dobrodošlicu za korisnika sa porukom velikim slovima ali ne preterano animirano jer ne treba da korisniku odvuče pažnju. Pošto početna strana treba samo da služi da dočeka korisnika, a smisao i zadatak ovog dela aplikacije je da korisnik kupi nešto, na njoj treba da se nalaze proizvodi koje sama prodavnica izdvaja iz ponude. Ti proizvodi su ili oni koji su na popustu, ili su pri kraju zaliha, ili je veoma popularno u zadnje vreme pa možda bi nešto od toga kupca zanimalo da kupi.

4.2.2.2

Galerija u sebi sadrži skup slika koje predstavljaju izgled prodavnice. Slike su raspoređene po sortirane po datumu objavljivanja. Svakoj slici se može pristupiti u *fullscreen* modu kako se može videti u punoj veličini i moguda je navigacija slika u *fullscreen* modu prelaskom na predhodnu i sledeđu sliku. Time je omogudeno korisniku da ima uvid o tome kako izgleda sama prodavnica u kojoj želi da kupi stvari. Te slike mogu (ali ne bi trebale) da predstave i reklamiraju konkretne proizvode sa cenama. U bazi podataka se trebaju čuvati samo putanje do slika, ali ne i same slike.

4.2.2.3 Kontakt

Neretka je situacija da korisnik ima neka pitanja vezana za neki konkretan artikl ili bilo šta vezano za prodavnicu koja možda nisu odgovorena na sajtu ili su mu neke stvari nejasne. Na *Contact* stranici ovog dela aplikacije nalaze se sve potrebne informacije kako bi korisnik kontaktirao vlasnike/radnike prodavnice.

4.2.2.4 Recenzije

Svako ko je nov na web aplikaciji može biti skeptičan oko njenog korišćenja. Preporuka od strane drugih korisnika je uvek dobra stvar. Zbog toga je ovde uvedena sekcija *review* za recenzije koje predstavljaju ocene i komentare drugih korisnika koji su koristili sajt i njegove usluge. Svaki ulogovani korisnik može da ostavi komentar i da oceni ocenom od 1 do 5.

4.2.2.5 Pretraga

Pretraga je jedan od najvažnijih delova samo aplikacije jer posredstvom nje, korisnik dolazi do željenog proizvoda. Napravljena je na više načina tako da korisnik može da bira onaj način koji je njemu optimalan: Klasična pretraga koja se sastoji iz jednog tekstualnog polja gde korisnik unosi izraz (jedna ili više reči) na osnovu kog se vrši pretraga, proširena pretraga koja nudi korisniku opciju da sam bira kriterijume pretrage. Nudi mu se iz izbora klasa i grupa opcije i kriterijume koje može da odredi i koje želi da uključi u pretragu.

4.2.2.7 Kupovina putem aplikacije

Svaki korisnik ima svoju "korpu za kupovinu". Kada se odluči da želi da kupi neki proizvod on prvo mora da je

doda u korpu. Ta korpa sama po sebi može biti toliko komplikovana da se može razviti kao posebna aplikacija koja je preko mikroservisa povezana za glavnim *backend*-om. Nakon kupovina korisniku se obavezno prikazuje odgovarajude obaveštenje da je kupovina uspešno izvršena (ili ako je došlo do neke greške da ga obavesti o tome). Obavezno se korisniku šalje mejl sa celim izveštajem i računom o kupovini sa podacima o tome šta i kada je kupio, kako i koliko je platio.

4.2.3 Kursevi

Deo aplikacije koji se bavi kursevima sadrži uputstvo za korisnika kako da dođe do nekog predavača koji bi ga naučio da svira neki instrument ili da savlada neku veštinu sviranja (*arpeggio, blast beat...*). Kao i svaki deo ove aplikacije, sastoji se od početne stranice koja sadrži dobrodošlicu i kratko uputstvo kako da dođe do predavača koji bi bio optimalan za njega.

4.2.3.1 *Online* kursevi

Ukoliko se korisnik odlučio da gleda kurseve preko aplikacije (jer smatra da mu je tako lakše da nauči) poseduje ovaj segment aplikacije. Kursevi prestavljaju video klipove u kojima jedna ili više osoba objašnjavaju korisniku načine i korake koji su potrebni da bi savladao određenu tehniku. Mana ovog načina jeste to što korisnik nema nikakvu interakciju sa predavačem. Zbog toga ovi video klipovi moraju biti jasni i razumljivi.

4.2.3.2 Uživo kursevi

Nekim korisnicima je lakše interaktivno da ih neko nauči nešto novo. Zbog toga, aplikacija nudi kurseve uživo sa predavačima koji se mogu rezervisati. Pretraga za *online* kurs i za rezervaciju predavača funkcionišu na sličan način samo što ovde još korisnik dodatno naglašava dane i vremenske opsege u nedelji kada on želi da ide na časove.

4.2.3.3 Sekcija tutorijala

Aplikacija nudi posebnu sekciju gde korisnici mogu da gledaju besplatne video klipove koji im mogu pomodi oko nekih sitnijih stvari. Ovaj deo je tu da početnicima demonstrira neke proste stvari koje su opšte dostupne i da ih ovde lakše pronađu. Radi konzistentnosti, ovde takođe postoji pretraga na osnovu instrumenata i oblasti samo što rezultujući video klipovi nisu "zaključani" i svako ih može pogledati, oceniti i prokomentarisati.

4.2.4 Servis

Nije retka stvar da se neki instrument može pokvariti, polomiti, vremenom istrošiti ili imati bilo kakav drugačiji kvar. Za takve situacije postoje majstori koji se bave takvim problemima odnosno njihovim rešavanjem. Definisati kvar nije lako objasniti preko bilo koje aplikacije pa je ovaj malo detaljnije napravljen kako bi korisnik najlakše identifikovao kvar. Nakon pretrage korisniku se nudi spisak majstora i osoba koji obavljaju popravku i otklanjanje kvarova.

5. ZAKLJUČAK

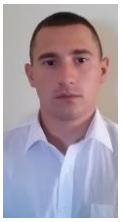
U današnjem modernom vremenu, sve više i više stvari su digitalizovane. Sve više i više stvari se obavljaju posredstvom mobilnih telefona, kompjutera ili drugih elektronskih uređaja. *Web* tehnologije su postale sve više i više zastupljenije jer ne zahtevaju nikakav dodatan napor pri instaliranju nekih programa i dodataka (osim rutinske instalacije u „3 klika"). Jako se lako koriste. Aplikacija „Trio Music" omogućava tri stvari koje ranije nisu mogle biti obavljene na jednom mestu na toliko prost način. Korisnik samim prijavljivanjem na sajt dobija mogućnost da pregleda artikle u samoj prodavnici što mu znatno skraćuje vreme da luta po raznim prodavnicama koje možda i nemaju ono što on traži ili mu cena ne odgovara ili je najbliža muzička prodavnica previše udaljena pa mu online kupovina omogudava i lakše i brže pladanje.

Ne mora da „kopa" po internetu da bi pronašao kurseve koji mu možda odgovaraju, a možda i ne. Ovde ima sve lepo na jednom mestu uz mogućnost uvida recenzija prethodnih korisnika. Ne mora da pretražuje silne stranice i kataloge kako bi pronašao servisera da otkloni neki kvar koji mu se desio. Ovde takođe može lako da ih pronađe bez većih poteškoda (ukoliko ne pronađe iz prve majstora, administratori mu pomažu oko toga).

6. LITERATURA

[1] *Client/Server applications*
https://en.wikipedia.org/wiki/Client%E2%80%93server_model
[2] *Backend*
https://learntocodewith.me/posts/backend-development/
[3] *SOAP*
https://www.tutorialspoint.com/soap/what_is_soap.htm
[4] *REST*
https://www.codecademy.com/articles/what-is-rest
[5] *Frontend*
https://en.wikipedia.org/wiki/Front-end_web_development
[6] *HTML*
https://www.w3schools.com/html/
[7] *JavaScript*
https://en.wikipedia.org/wiki/JavaScript
[8] *MySQL*
https://en.wikipedia.org/wiki/MySQL

Kratka biografija:



Vladimir Jovičić rođen je u Somboru 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Računarstvo i automatika - Primenjene računarske nauke i informatika - Elektronsko poslovanje odbranio je 2019.god.

kontakt: vlada.jova@yahoo.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE12Budnic

OTKRIVANJE SIGURNOSNIH PROPUSTA U *SCADA* SISTEMIMA METODOM *FUZZ* TESTIRANJA

IDENTIFYING SECURITY VULNERABILITIES IN *SCADA* SYSTEMS VIA *FUZZ* TESTING METHOD

Spasoje Budnić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu opisani su problemi koji nastaju usljed propusta u razvoju programa. Objašnjena je fuzz metoda za pronalaženje sigurnosnih propusta. Fuzz metode se relativno često koriste u rasprostranjenim IT sistemima kao što su web sistemi, ali u slučaju SCADA sistema, gde postoje razni protokoli uključujući i vlasničke, ne postoji uniformno i dostupno rešenje za fuzz testiranje. S toga je cilj rada da istraži pristupe i predloži proširenje postojećih platformi tako da se omogući i SCADA fuzz testiranje.*

Ključne reči: *SCADA sigurnost, Modbus protokol, fuzz testiranje*

Abstract – *This paper describes the problems that arise from vulnerabilities in program development. The fuzz method for finding security leaks is explained. Fuzz methods are relatively commonly used in widespread IT systems such as web systems, but in the case of SCADA systems, where there are various protocols, including proprietary ones, there is no uniform and accessible solution for fuzz testing. Therefore, the aim of the paper is to explore approaches and propose extensions to existing platforms to enable SCADA fuzz testing.*

Keywords: *SCADA security, Modbus protocol, fuzz testing*

1. UVOD

Veliki broj sigurnosnih propusta u softveru nastaje zbog lošeg rukovanja ulazima ili zbog grešaka u logici programa. *SCADA* sistemi su postali izuzetno važni za većinu industrija širom svijeta jer kontrolišu kritične infrastrukture [1] kao što su električne mreže, vodovodi, gasovodi...

Stariji tipovi ovih sistema su koristili svoje izolovane mreže za komunikaciju [2]. Sistemi su uglavnom bili specifični po tome koju opremu i protokole koriste.

SCADA sistemi su izloženi internet prijetnjama, kao i ostali informacioni sistemi. Napadom na *SCADA* sisteme, postoji mogućnost da dođe do fizičkog uništenja stvari [1,3], što može biti zloupotrijebljeno od strane terorista. Mnogo je napada zabilježeno poslednjih nekoliko godina na industrijske sisteme.

Fuzz testiranje je tehnologija koja pokušava da otkrije sigurnosne ranjivosti slanjem nasumičnih ulaza aplikaciji ili uređaju [2, 4].

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Branislav Atlagić, docent.

2. *SCADA*

SCADA (Supervisory Control And Data Acquisition) podrazumijeva nadzor, kontrolu i prikupljanje podataka. Tradicionalni sistemi su se sastojali od jednog *PLC*-a koji je kontrolisao industrijsko postrojenje [5]. Vremenom se javila potreba za upravljanjem sistemima sa distribuiranim resursima pa je *SCADA* prerasla u distribuirani sistem. Danas su *SCADA* sistemi prisutni u skoro svim kritičnim infrastukturama [1]. *SCADA* sistem je zapravo jedan akviziciono-upravljački sistem. Akvizicija podataka podrazumijeva dobavljanje vrijednosti sa digitalnih ili analognih mjerila iz polja. Akvizicija je neizostavan proces u svim *SCADA* sistemima jer se sve odluke donose na osnovu prikupljenih i obrađenih podataka [5].

Kada su ovi sistemi prvobitno razvijani, vrlo malo pažnje se posvećivalo zaštiti od zlonamernih napada. Sistemi su bili fizički izolovani, koristili su vlasničku opremu, kao i specifične komunikacione protokole. Iz dobro poznatih razloga, ovakvi sistemi su sada pogodni za mrežne napade. Nedostatak autentifikacije pošiljaoca, korišćenje komercijalnih operativnih sistema, upotreba podrazumijevanih lozinki ili odsustvo lozinki za pristup uređajima samo su neki od razloga zašto ovi sistemi mogu biti laka meta [2].

3. INDUSTRIJSKI PROTOKOLI

Industrijski protokoli predstavljaju grupu komunikacionih protokola specifičnih za *SCADA* sisteme. Tradicionalno, industrijski protokoli su često vezani za specifičnu oblast primjene, još češće za proizvođača opreme [5]. U Americi je dominantan *DNP3* protokol, dok su u Evropi *IEC 60870-5-101* i *IEC 6087-5-104* [6]. U starijim sistemima još uvijek se koriste *Modbus, Fieldbus,* kao i mnogi drugi vlasnički protokoli [6].

4. MODBUS

Modbus protokol razvijen je 1979. godine od strane *Modicon* kompanije. Protokol i specifikacija su potpuno otvoreni što je u velikoj mjeri doprinijelo njegovoj popularnosti [2, 5].

Modbus-ASCII i *Modbus-RTU* [5] su dvije prvobitne implementacije protokola za rad preko serijskog *UART* kanala, tekstualna i binarna verzija. S pojavom *ethernet*-a razvijaju se verzije *Modbus-TCP* [2] i *Modbus-UDP* koje predstavljaju enkapsulaciju *Modbus-RTU* protokola preko *TCP* i *UDP* protokola.

Tabela 1. *Struktura Modbus poruke*

	MBAP				*PDU*	
	Transaction ID	*Protocol ID*	*Length*	*Unit ID*	*Function Code*	*Payload*
Veličina u oktetima	2	2	1	1	1	N

U tabeli 1 se vidi struktura *Modbus-TCP* poruke. Poruka se sastoji od zaglavlja (*MBAP)* i tijela (*PDU).*

Transaction ID predstavlja identifikator transakcije. Zahtjev i odgovor treba da imaju identičan identifikator transakcije. *Master* stanica generiše identifikator po nekom internom algoritmu, dok *slave* stanica treba da vrati odgovor sa istim identifikatorom.

Protocol ID je identifikator protokola. U ovom slučaju je uvijek 0.

Length označava broj okteta koji slijede poslije njega.

Unit ID je identifikator stanice, koristan je kada se više *slave* stanica nalazi iza *hub*-a. *Hub* proslijeđuje svima poruku, pa je identifikator stanice jedini način da se odredi kome poruka pripada.

Function Code je kôd funkcije koju udaljena stanica treba da izvrši. Veličine je jednog okteta pa je broj mogućih vrijednosti od 0 do 255, međutim u slučaju zahtjeva validne vijednosti su od 0 do 127 [2]. Vrijednosti od 128 do 255 koriste se ukoliko zadata funkcija nije uspješno izvršena. Tada udaljena stanica vraća odgovor gdje je *FunctionCode* zahtjeva uvećan za 0x80.

Modbus uređaji i simulatori ne moraju da podržavaju sve funkcije definisane protokolom, međutim većina njih bi trebalo da podržava funkcije za manipulisanje digitalnim i analognim registrima (tabela 2).

Tabela 2. *Osnovne Modbus poruke*

Function code	*Modbus message*
1	*Read Coils*
2	*Read Discrete Inputs*
3	*Read Holding Registers*
4	*Read Input Registers*
5	*Write Single Coil*
6	*Write Single Register*

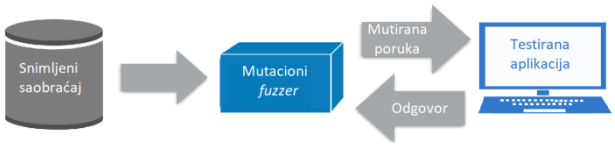
5. *FUZZ* TESTIRANJE

Fuzz testiranje je sigurnosna tehnika gdje se kreiraju ulazi koji se proslijeđuju ciljanom softveru sa namjerom da se pronađu nedostaci ciljanog programa. Najčešći nedostatak koji se uočava *fuzz* testiranjem je loša validacija ulaznih podataka. Nerijetko *fuzz* testiranje može da otkrije i nedostatke u aplikacionoj logici. Danas većina modernih programa ima dobru provjeru ulaza i može da obradi slučajno izabrane nizove okteta bez rušenja. Stoga se moderni alati za *fuzz* testiranje prilagođavaju modernim programima kako bi bili što efikasniji. Moderni alati biraju niz okteta na osnovu određene semantike. Obično postoji više slojeva aplikacije kroz koje podaci moraju proći prije nego što dođu do same aplikacione logike [4].

5.1 Vrste *fuzzer*-a

Na osnovu kreiranih testnih podataka, *fuzzer*-i se dijele u dvije grupe: generacioni i mutacioni.

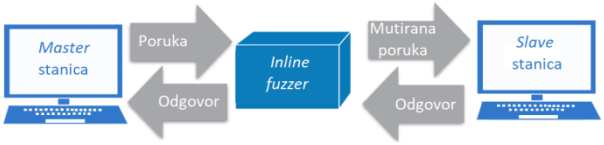
Generacioni *fuzzer*-i generišu testne ulaze na osnovu pravila kako treba da izgleda pravilan ulaz [3]. Najjednostavniji generacioni *fuzzer*-i kreiraju ulaze sa slučajnim odabirom.



Slika 1. *Mutacioni fuzzer*

Mutacioni *fuzzer*-i (slika 1) funkcionišu tako što čitaju poznate ulaze u koje ubacuju *loše* oktete i na taj način kreiraju testne ulaze [3]. Većina mutacionih *fuzzer*-a koristi prethodno snimljeni saobraćaj kao polaznu osnovu za mutiranje ulaza. *GPF (General Purpose Fuzzer)* [4] ne zahtjeva poznavanje protokola ili strukture podataka. Koristi jednostavne heuristike da pogodi granice i mutira ulaze na osnovu njegovih pretpostavki. Stoga *fuzzer*-i ne moraju da poznaju strukturu ulaza koje *napadaju*. Mogu slijepo da čitaju *dobre* ulaze i da mutiraju bez poznavanja semantike protokola [4].

Nedostatak ovakvih *fuzzer*-a je što su u stanju da testiraju serversku stranu, dok klijenti ostaju netestirani. Nakon kreiranja paketa, šalju ih na specificiranu adresu i port. Klijenti ne prihvataju saobraćaj kao serveri pa samo alati koji prate mrežni saobraćaj mogu da testiraju klijente.



Slika 2. *Inline fuzzer*

Inline fuzzer-ima (slika 2) se nazivaju alati koji prate mrežni saobraćaj. Koriste tehniku presretanja *Man in the Middle* kako bi presreli mrežne podatke. U kritičnim infrastrukturama veliki broj protokola radi sa vremenski osjetljivim sesijama pa se ne mogu testirati sa alatima koji zahtijevaju veliko skladištenje paketa, već se koriste *inline* alati [4].

5.2 Testiranje vlasničkih protokola

Vlasnički protokoli koje koristi *SCADA* oprema najčešće nisu dobro razumljivi. Domenski eksperti najčešće nemaju dovoljno vremena ili sposobnosti da izvrše testiranje nad protokolima koje oprema koristi [4].

S druge strane, eksperti koji se bave *fuzz* testiranjem mogu obaviti posao umjesto domenskih eksperata. Problem nastaje u kritičnim infrastrukturama gdje su domenski eksperti veoma oprezni i ne dozvoljavaju strancima da rade sa njihovom opremom [4].

LZFuzz se koristi za testiranje vlasničkih *SCADA* protokola gdje je gramatika nepoznata. Pronalazi tokene poruka koristeći *Lempel-Ziv*-ov algoritam. *LZFuzz* koristi informacije iz tokena kako bi mutirao poruke [7].

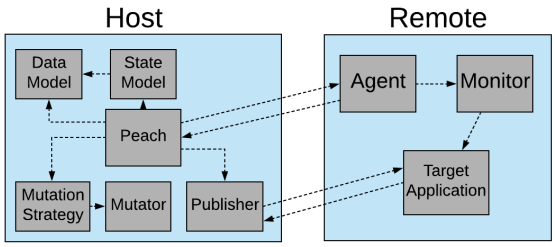
Međutim, ovo je *fuzz* tehnika zasnovana na mutaciji, pa je teško primijeniti na protokol sa komplikovanom ulaznom gramatikom. Stoga većina alata za *fuzz* testiranje *SCADA* protokola trenutno koristi gramatiku protokola [3].

6. *PEACH FUZZER*

Peach je platforma za automatsko testiranje i prevenciju napada pronalaženjem propusta u sistemima [8]. Koristi tehniku bijele kutije za testiranje programa. *Peach Community Edition* je softver otvorenog koda koji je korišten za testiranje *Modbus* protokola u ovom radu.

Peach odvaja model podataka od samog sistema koji se testira, kao i od samog alata. Ovaj alat pruža sistem za praćenje i detektovanje grešaka. Alat je veoma fleksibilan i proširiv. Ukoliko je potrebno testirati neki specifičan sistem, moguće je dodati nove komponente kako bi se postigla veća efikasnost.

Peach pruža mogućnost kreiranja različitih *fuzzing* strategija, kao i kreiranje specifičnih mutatora podataka. Kombinacija ovoga omogućava potpunu kontrolu nad testiranjem. Jednostavna proširivost alata omogućava testerima da urade različita testiranja, a da pri tom ne prave novi alat.



Slika 3. *Arhitektura Peach platforme*

Na slici 3 prikazana je arhitektura *Peach fuzzer*-a. Većina modernih *fuzzer*-a teži sličnoj arhitekturi koja omogućava brzu i laku adaptaciju. Slika 3 prikazuje najčešći slučaj gdje se testirana aplikacija nalazi na udaljenom računaru. Prilikom testiranja aplikacija opisanih u ovom radu, *fuzzer* i testirana aplikacija se nalaze na istom računaru.

Model definiše model podataka koji se testira. U ovom slučaju je definisan *Modbus* model podataka. Najviše vremena treba posvetiti definisanju modela podataka. U zavisnosti od toga kako je modelovan protokol, zavisi i uspješnost samog testiranja.

Izdavač (*Publisher*) predstavlja ulazno-izlazni *interface*. Izdavači pružaju stvarni transport i realizaciju. Primjer ugrađenih izdavača su *TCPPublisher*, *UDPPublisher*, *ConsolePublisher*... U ovom radu testiran je *Modbus* protokol koji se prenosi preko *TCP* protokola, pa je moguće koristiti *TCPPublisher*. Za *Modbus-RTU* protokol neophodno je kreirati novog izdavača.

Strategija testiranja je logika koja predstavlja na koji način će se modifikovati elementi modela podataka. Strategija određuje koji mutatori će se koristiti i na koji način, kao i da li će se modifikovati neki dio podataka ili ne. Sekvencijalna i nasumična su neke od ugrađenih strategija.

Mutator se koristi za kreiranje podataka. Mutatori najčešće koriste postojeće zadate vrijednosti koje mutiraju. Mutatori mogu kreirati nove elemente, kreiraju niz zadate dužine sa slučajno izabranim elementima.

Agenti su posebni procesi koji se mogu pokrenuti lokalno ili na udaljenom računaru i pokreću jedan ili više monitora. Zadatak monitora je praćenje nekog procesa.

6.1 *Peach Pit*

Peach Pit file je *XML file* koji sadrži neophodne informacije za testiranje. *Peach Pit* je konfiguracioni dokument koji sadrži informacije o modelu podataka koji se testira, kao i o agentima i monitorima koji se koriste prilikom testiranja.

6.2 *Modbus* model podataka

Model bi trebalo da predstavlja validnu gramatiku protokola. *Peach fuzzer* koristi tehniku bijele kutije baziranu na gramatici protokola, nastojeći da pronađe propuste u programu.

Osnovni cilj prilikom testiranja aplikacije je pogoditi granične slučajeve i pri tom smanjiti broj testnih slučajeva koji će odmah biti odbačeni. Klasičan primjer su protokoli koji sadrže sigurnosnu provjeru, kao što su *CRC* ili *LRC*. Ukoliko ova provjera nije validna, vjerovatnoća da testni podaci ne budu odbačeni već prilikom prve validacije je minimalna. U slučaju *Modbus-TCP* protokola sigurnosna provjera se nalazi u sklopu *TCP* protokola.

Da bi testirali logiku aplikacije, podaci moraju biti *skoro* validni, odnosno neophodno je da prođu ulazne validacije i dopru do same logike.

Jedan od načina kreiranja podataka je da se u svakom testnom slučaju šalje validno zaglavlje. Prilikom kreiranja modela treba voditi računa da uređaji i sistemi ne moraju da podržavaju sve protokol funkcije, pa bi kreiranje testnih slučajeva za funkcije koje nisu podržane smanjilo efikasnost pronalaženja propusta.

U ovom radu objašnjena su polja funkcija iz tabele 2. Prilikom testiranja aplikacija, pored ovih, testirane su i funkcije *WriteMultipleCoils* i *WriteMultipleRegisters.*

U ovom primjeru mutirana su samo polja koja se nalaze poslije koda funkcije u *Modbus* poruci: *Start Address*, *Quantity*, *Output Address* i *Value.* Sva polja su mutirana tako da im se postavljaju najmanja i najveća vrijednost kao i nasumično izabrane vijednosti. Takodje korišteni su mutatori koji dupliraju elemente ili prave nizove elemenata.

Testiranjem graničnih vrijednosti i dupliranjem elemenata se mogu izazvati prelivanje *buffer*-a i aritmetičko prelivanje. Prelivanje *buffer*-a se dešava prilikom zapisivanja podataka van granica *buffer*-a. Može se iskoristiti da se zamijeni povratna adresa na *stack*-u i pokrene izvršavanje zlonamernog koda. Aritmetičko prelivanje nastaje kada rezultat aritmetičike operacije prelazi granice promjenjive u koju treba da sačuva rezultat.

6.2.1 *Start Address* polje

Start Address polje predstavlja adresu prvog registra u funkcijama za dobavljanje vrijednosti. Funkcije koje sadrže ovo polje su *Read Coils*, *Read Discrete Inputs*, *Read Holding Registers* i *Read Input Registes*.

6.2.2 *Quantity* polje

Quantity polje u funkcijama za dobavljanje vrijednosti označava koliko registara je potrebno dobaviti. Nalazi se poslije *Start Address* polja. Funkcije koje sadrže ovo polje su *Read Coils, Read Discrete Inputs, Read Holding Registers, Read Input Registers, Write Multiple Coils* i *Write Multiple Registers.*

6.2.3 *Output Address* polje

Output Address polje označava adresu registra u koji je potrebno upisati vrijednost. Funkcije koje sadrže ovo polje su *Write Singe Coil i Write Single Register*.

6.2.4 Value polje

Value polje označava vrijednost koju je potrebno upisati u registar. Funkcije koje sadrže ovo polje su: *Write Singe Coil* i *Write Single Register.*

7. PRIMJENA NAD REALNIM APLIKACIJAMA

Testirane su sledeće *Modbus* aplikacije otvorenog koda: *MOD_RSSIM, PyModSlave* i *EasyModbusTCP* server simulator. Na sve testirane aplikacije primjenjen je model podataka opisan u poglavlju 6.2. Korištene su sekvencijalna i nasumična strategija sa najviše 10000 iteracija. Uočene su određene anomalije u ponašanju aplikacija.

7.1 *MOD_RSSIM*

MOD_RSSIM je simulator *SCADA* udaljene stanice. Prilikom testiranja ovog simulatora uočeno je da poslije određenog vremena simulator prestaje da odgovara na zahtjeve. S obzirom da je simulator testiran u iteracijama, na početku svake iteracije se uspostavlja veza sa simulatorom, a na kraju iteracije prekida. U ovom slučaju desio se *DoS (Denial of Service)* napad. Jedan od slučajeva kada se desi *DoS* je slanje samog zaglavlja u iteracijama.

7.2 *PyModSlave*

Prilikom testiranja *PyModSlave* simulatora uočeno je da već na početku testiranja simulator postaje nedostupan za akcije sa korisničkog interfejsa. Simulator komunicira sa *fuzzer*-om, ali nije moguće izvršiti nijednu komandu putem korisničkog interfejsa.

7.3 *EasyModbusTCP* server simulator

Prilikom testiranja *EasyModbusTCP* server simulatora, *windows debugger* je registrovao prestanak rada aplikacije veći broj puta. Ukoliko monitor, u ovom slučaju *windows debugger*, registruje nevalidno stanje, *fuzzer* pokušava da reprodukuje to stanje ponovo, kako bi verifikovao sekvencu koja je to stanje izazvala.

Jedna od grešaka je aritmetičko prelivanje kada je kôd funkcije veći od 127. Naknadnom forenzikom je utvrđeno da nema provjere da li je kôd funkcije validan. Prilikom postavljanja greške da funkcija nije podržana dolazi do aritmetičkog prelivanja.

Poruka *WriteMultipleCoils* može *srušiti* simulator ukoliko je *Quantity* veći od stvarnog broja poslatih registara.

6.1 Verifikacija uspjeha

Ukoliko se prilikom testiranja *SCADA* sistema uoči bilo kakva anomalija u ponašanju, može se smatrati da je testiranje bilo uspješno. Ukoliko se prilikom testiranja ne dokaže nijedan propust u implementaciji, to nije dokaz da propusti ne postoje. Možda samo testni slučajevi nisu dobro kreirani.

Ovim radom je ustanovljeno da u izvornom kodu programa opisanog u poglavlju 7.3 treba dodati provjeru kako bi se spriječilo aritmetičko prelivanje. Takođe treba dodati i provjeru da li prilikom *WriteMultipleCoils* zahtjeva primljeni broj vrijednosti digitalnih registara odgovara vrijednosti polja *Quantity.*

8. ZAKLJUČAK

Osnovna zamisao ovoga rada je ukazivanje na sigurnosne propuste u *SCADA* sistemima nastale lošim rukovanjem ulazima, kao i propusti unutar same logike.

Metod *fuzz* testiranja postaje sve popularniji zbog brzine u pronalaženju grešaka. Moderni alati koriste gramatiku za kreiranje ulaza i postižu efikasnije rezultate od alata koji nasumično biraju ulaze.

Za realizaciju rada bilo je neophodno poznavanje *Modbus* industrijskog protokola, kao i upoznavanje sa tehnikom *fuzz* testiranja.

Za sprovođenje testiranja izabran je *Peach Community Edition* alat otvorenog koda, koji se relativno lako modifikuje i konfiguriše da radi sa *Modbus* protokolom.

U ovom radu predstavljeno je testiranje *Modbus* serverskih aplikacija, a unaprijeđenje alata bi podrazumijevalo testiranje i klijentske strane.

8. LITERATURA

[1] Kyle Coffey, Richard Smith, Leandros Maglaras and Helge Janicke, *Vulnerability Analysis of Network Scanning on SCADA Systems*, 2018

[2] Mehdi Sabraoui, Jeffery L. Hieb, and James H. Graham, *Protocol Fuzzing for Cyber Security and Hardening of Industrial Control Systems*, 2014

[3] Hyunguk Yoo, Taeshik Shon, *Grammar-based Adaptive Fuzzing: Evaluation on SCADA Modbus Protocol*, 2016

[4] Rebecca Shapiro, Sergey Bratus, Edmond Rogers, Sean Smith, *Do it yourself SCADA vulnerability testing with lzfuzz*, 2011

[5] Branislav Atlagić, *Softver sa kritičnim odzivom, projektovanje SCADA sistema*, 2015

[6] Frances Cleveland, *IEC TC57 Security Standards for the Power System's Information Infrastructure – Beyond Simple Encription*, 2006

[7] Sergey Bratus, Axel Hansen, Anna Shubina, *LZfuzz: a fast compression-based fuzzer for poorly documented protocols*, 2008

[8] Peach alat *(https://www.peach.tech/products/peach-fuzzer/*) (pristupljeno u septembru 2019)

Kratka biografija:

Spasoje Budnić rođen je 1995. godine u Trebinju. Završio je Srednju Tehničku Školu u Trebinju 2014. godine. Fakultet tehničih nauka u Novom Sadu upisao je 2014. godine. Diplomski rad pod nazivom *Akvizicija podataka u SCADA sistemima upotrebom Series 5 protokola* odbranio je 2018. godine.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE13Banjanin

PARADIGMIČKA I FUNKCIONALNA ANALIZA NODE.JS PLATFORME

PARADIGMIC AND FUNCTIONAL ANALYSIS OF NODE.JS PLATFORM

Una Banjanin, Srđan Popov, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu prezentovani su osnovni koncepti i karakteristike NodeJs platforme. Takođe, prikazana je i uopšteno skript paradigma kao i sami aspekti paradigme. Implementirana aplikacija Super Chat je jednostavna web aplikacija za razmenu poruka.*

Ključne reči: *NodeJs, Objektno-orijentisana paradigma, JavaScript, AJAX*

Abstract – *This thesis is presented by the basic concepts and characteristics of the NodeJs platform. There is also presented script paradigm as well as the aspects of the paradigm. Implemented Super Chat application is a simple web application for chat.*

Keywords: *NodeJs, Object-oriented paradigm, JavaScript, AJAX*

1. UVOD

Node.Js (takođe nazvan i Node) je platforma izgradjena na Google Chrome V8 izvršnoj mašini za lako pokretanje brzih, skalabilnih i laganih aplikacija [1].

U ovom radu se bavimo kako NodeJs-om, tako i samom skript paradigmom kao i njenim aspektima. Glavna karakteristika Node-a je njegova upotreba neblokirajućeg I/O vođenog događaja sa asihronim modelom programiranja koji ostaju efikasni u upravljanju konkurentnosti. Node se razlikuje od JavaScripta koji opisujemo, naglašavajući neke nedostatke koje Node pokriva. Isto tako predstavljajući AJAX sa njegovim prednostima i manama, pokazujemo kako Node nadmašuje AJAX što se tiče upotrebljivosti razvojnih aplikacija u realnom vremenu.

2. SKRIPT PARADIGMA

2.1 Šta je skriptni jezik

Skriptni jezik je oblik programskog jezika koji se koristi za kreiranje skripti ili bita koda. Jezici za pisanje skripti često se pišu kako bi se olakšale i poboljšale karakteristike internet stranice.
Te karakteristike se obrađuju na serveru, ali se skripta određene stranice pokreće u internet pretraživaču korisnika.
Jezik za pisanje skripti kontroliše rad normalno interaktivnog programa, dajući mu redosled rada za sve „na jednoj gomili".

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

2.2 Prednosti i nedostaci skriptnog jezika

Što se tiče samih prednosti ovog jezika, navešćemo nekoliko značajnih: Lako se uči i koristi, potrebno je minimalno znanje ili iskustvo u programiranju, omogućava izvođenje složenih zadataka u relativno malom broju koraka.
Nedostaci su ti što može doći do sporijeg pokretanja budući da su kodovi interpretirani i da nisu kompajlirani u mašinski kod. Takođe, ukoliko dođe do grešaka, one mogu biti teže za otklanjanje jer nije dostupno izvorno razvojno okruženje.

2.3 Zašto moderne internet stranice koriste JavaScript

Podrška za pretraživač, Korišćenjem JavaScript-a možemo dodati različite funkcije kao što su autentifikacija korisika, validacija korisnika, itd. JavaScript lako čita i piše HTML elemente i može se lako ugraditi u HTML, akcioni događaj se može kreirati korišćenjem JavaScripta u trenutku kada korisnik pritisne dugme.

2.4 Zašto koristiti skriptne jezike

Vrlo često želimo automatizovati jednostavan zadatak - pokretanje nekoliko programa u nizu, instaliranje programa, pa čak i pisanje jednostavnog skripta ili GUI-a za pokretanje programa koji zahteva mnogo parametara. Upravo skriptni jezici nam dozvoljavaju da tako nešto brzo napišemo i pokrenemo bez kompajliranja. Takođe imaju dobru podršku za pokretanje procesa i njihovo kontrolisanje.

2.5 Klijentska i serverska strana

Na strani klijenta, skriptovanje vrši svu računicu na korisničkom računaru. Internet pretraživač ili određeni dodatak čita skriptu i pretvara je u vizuelnu internet stranicu. **Front end** kontekst koristi skriptovanje kroz korisnički interfejs. Većina informacija koje korisnik unosi u internet stranicu ostraje na strani klijenta i ponekad se vraća na server. JavaScript može dobiti spoljne slike i datoteke.

3. PREDSTAVLJANJE NODE.JS

Node.Js (Node) je okruženje koje je prvobitno razvijeno 2009. godine od strane Riana Dahla za razvoj aplikacija na strani servera.

Može se smatrati serverskim JavaScriptom. Napravljen je da se bavi problemima koje platforme mogu imati sa performansama u mrežnim komunikacijama – konkretno misleći na prekomerno vreme obrade internet zahteva i odgovora na isti.

Node je zasnovan na događajima, a ne na nitima. Koristi petlju događaja u okviru jedne niti umesto više niti i može da se skalira na milion istovremenih veza.

3.1 Ciljevi studije

Ubrzo nakon što je nastao WWW(World Wide Web), stvoren je i JavaScript. Kao takav, igrao je ključnu ulogu u dodavanju interakcije u korisnički interfejs internet aplikacija i internet stranica do izdanja HTML5 i modernih JavaScript okvira**.** Uprkos ovom napretku, JavaScript je smatran jezikom skriptovanja za programiranje na strani klijenta, koji kao takav, radi isključivo iz internet pretraživača. Međutim, ovaj pristup promenio se sa razvojem serverskog JavaScripta (među kojima je istaknut Node).

3.2 Pravljenje servera pomoću Node-a

Jedna od uobičajenih upotreba Node-a je za izgradnju servera. Može se koristiti za kreiranje različitih tipova servera.

3.3 Node arhitektura

Node.Js se sastoji od više komponenti. V8 je JavaScript mehanizam. Libeio obrađuje unutrašnji bazen koji se koristi da bi sinhroni POSIX pozivi postali asinhroni u petlji događaja. Asinhroni sistem datoteka se ne koristi, već se blokiranje I/O obavlja u njegovoj vlastitoj niti, takav da je ne-blokirajući u petlji događaja u Node.Js-u. Libev je petlja događaja. [2]

Node.Js koristi tehnike kao što su **kqueue, select, epoll** ili **/dev/poll** za dobijanje obaveštenja iz operativnog sistema kada dolaze nove konekcije, zatim otpremljuje nove događaje programeru, koji dalje rukuje njima.

3.4 Node.Js Moduli

Moduli predstavljaju dodatke i proširenja za Node koji pomažu u razvojnom procesu. Node modul izlaže javni API(Interfejs za programiranje aplikacija) koji se može koristiti nakon uvoza modula u trenutni skript. Mogu se kategorizovati kao jezgreni moduli, moduli trećih strana i lokalni moduli. Osnovni moduli su moduli koji dolaze sa Instalacijom Node-a i prethodno su učitani kada se pokrene Node proces.

3.5 Express Modul

Express je Node modul koji pruža minimalan i fleksibilan okvir za Node.js internet aplikacije. Obezbeđuje robusne i čiste funckije koje se dodaju Node modulima. Express se instalira koristeći NPM paket mendzer koji izdaje "npm install express" komandu. Express server se sastoji od tri bloka: *Rutera ,Rute I Middleware-a.* Ruta u Express-u je kombinacija HTTP glagola i putanje. HTTP glagol je obično jedan od četiri HTTP metode: GET, POST, PUT i DELETE, a putanja je lokacija resursa (URI).

3.6 Node Package Manager NPM

NPM je ugrađeni alat koji je podrazumevano uključen u svaku instalaciju Node-a. On pomaže u lakom upravljanju modulima u Node projektima preuzimanjem paketa, rešavanjem zavisnosti, pokretanjem testova, i instalacijom uslužnih programa komande linije.
NPM se pokreće iz prevodioca komandne linije (CLI) i upravlja svim zavisnostima Node aplikacije. NPM automatski obrađuje sve procese preuzimanja i čuva sve imenovane module u „node-moduli" folderu.

3.7 Princip rada Node.Js-a

Glavne karakteristike Node arhitekture su korišćenje neblokirajućeg događaja i asinhroni I/O pozivi koji rade u jednoj niti. Konvencionalni internet serveri rukuju konkurentno, praveći nove niti za svaki novi zahtev koje mogu zauzeti slobodnu memoriju do maksimuma. Čak i sa ograničenom memorijom, i jednom niti, Node može postići visok stepen konkurentnosti bez potrebe za vršenjem prebacivanja konteksta između niti.

3.8 Neblokirajuća petlja događaja

Node je neblokirajući u smislu da može da servisira višestruke zahteve. Konvencionalni model blokiranja teži da blokira naknadne zahteve koji se šalju na server kada se vrši obavljanje I/O operacija, kao što je npr. čitanje sadržaja iz baze podataka. Da ne bi došlo do blokiranja, Node koristi petlju događaja (event loop), softverski šablon (patern) – šema u kojoj se aktivira povratna funckija događaja u momentu kada se neka od akcija dogodi u programu.

3.9 Jednonitni model (Single Threaded)

Node je proces koji se pokreće u petlji događaja i koristi jednu nit za servisiranje zahteva. Kada Node aplikacija treba da izvrši operacije, ona šalje asinhroni zadatak na petlju događaja, registruje funkciju povratnog poziva, a zatim nastavlja sa obradom druge operacije. Petlja događaja prati asinhronu operaciju, izvršava zadati povratni poziv i kada se završi, vraća njegov rezultat aplikaciji.

3.10 Asinhrono programiranje

Dok neblokirajući deo Node-a omogućava da prihvati gotovo sve izvršene zahteve, asinhrono programiranje omoguća da se zahtevi izvršavaju učinkovito korišćenjem ograničenih ciklusa takta i memoriji koja je na raspolaganju svojoj jednonitnoj arhitekturi. Asinhronizacija je u korenu Node-a zato što su skoro svi API-ji izloženi preko Node modula asinhroni.

3.11 Rešavanje konkurentnosti

Jedan način je „mrešćenjem" child niti, koje dele isti adresni prostor kao glavna nit izvršenja. Svaka nit dodeljuje svoj stack, registre i programski brojač. Drugi način je preko petlje događaja. Kada se dogodi neki događaj, pridruženi handler je stavljen u red za izvršenje.

3.12 Programiranje vođeno događajem (Event Driven)

Programiranje zasnovano na događajima je paradigma koja se često koristi za interkativne aplikacije. Interakcija korisnika stvara događaje, a glavna petlja izvršava odgovarajuće rukovodioce događaja (Event handler-e).

Event handler-i u Node-u su obične funkcije koje se koriste kao povratni pozivi. Node.Js API metode često koriste funkcije povratnog poziva koje se na kraju izvršavaju, kada se završi ne-blokirajuća operacija (kao što je čitanje i pisanje). Petlja događaja u Node.Js prima signale završetka i izvršava povratni poziv.

4. POREDJENJE NODE.JS SA JAVASCRIPTOM

JavaScript je baziran na prototipu, objektno orijentisanom, labavo tipiziranom skriptovanju na strani klijenta. Radi isključivo unutar internet pretraživača. Koristi se za dodavanje interaktivnosti na internet stranice. Zbog toga je potrebna pomoć nekog drugog programskog jezika ukoliko mora da izvrši bilo kakvu interakciju sa serverom. I ako je Node zasnovan na JavaScriptu, i koristi konstrukciju JavaScripta za skoro sve svoje funkcionalnosti, on nudi potpuno drugačije okruženje od JavaScripta. Node se može smatrati nadskupom JavaScripta. Ima dodatne funkcionalnosti i funkcije pored svega što sadrži JavaScript.

4.1 Modularni sistem

Jedan od nedostataka u JavaScriptu je nedostatak modularnosti. Jedini način za povezivanje različitih skripti je korišćenjem drugih jezika, poput HTML-a. Umesto definisanja određenog broja globala, Node je uveo modularni sistem. Node obuhvata velik broj osnovnih modula kao što su *http, net, fs.*

4.2 Globalni objekat

Node implementira globale sa jasnim razdvajanjem. Koriste se sledeća dva globalna objekta: *Global* i *Process.*

4.3 Privremena memorija (Buffer)

Još jedan nedostatak u JavaScriptu je njegova podrška za rukovanje binarnim podacima. Node Buffer klasa je rešila ovaj nedostatak obezbeđujući API za jednostavnu manipulaciju podacima. Buffer je dodatak Node-u za četiri primitivna tipa podataka – *boolean, number, string.*

4.4 Poređenje Node-a i AJAX-a

Jedina sličnost između Node-a i AJAX-a je u tome što oba rade na JavaScriptu. Dok se Node uglavnom koristi za operacije na strani servera za razvoj kompletne serverske aplikacije, AJAX se koristi za operacije na strani klijenta za dinamički rad ažuriranja sadržaja stranice bez osvežavanja.

4.5 AJAX (Asinhroni JavaScript i XML)

AJAX je skup tehnika za kreiranje visoko interaktivnih internet stranica i internet aplikacija. Koristi se za upućivanje na sve metode komuniciranja sa serverom od klijenta koji koristi JavaScript. Koristeći AJAX, aplikacija može da pozove specifičnu procedure na serveru, i izvrši osvežavanje samo specifičnog dela internet stranice.

4.6 Razvoj aplikacija u realnom vremenu pomoću Socket.io modula

Web Sockets dozvoljava simultanu komunikaciju u oba smera, između klijenta i servera, ali u potpuno novom protokolu. Socket.io je biblioteka za internet aplikacije u realnom vremenu. Socket.io je vođen događajima i izlaže i komponente na strani klijenta i na strani servera. I klijentska i serverska strana u suštini čine istu stvar: dozvoljavaju slanje (ili emitovanje) događaja i pružaju način za definisanje rukovodioca događaja.

5. NODE.JS BEZBEDNOST

Porast potražnje za JavaScriptom u polju programiranja proširio se u opsegu od programiranja na strani klijeta do programiranja na strani servera. Kao rezultat toga, SSJS (Sever Side JavaScript) funckije su dostupne skoro svuda.

5.1 Skriptiranje na unakrsnim lokacijama (XSS)

XSS je napad koji omogućava napadaču da ubaci zlonamerni skript u internet aplikaciju. XSS ranjivosti su prouzrokovane neuspehom internet aplikacije da pravilno uradi validaciju unosa korisnika.

5.2 Odbijanje usluge (Denial of Servise - DoS)

DoS je napad koji informacije ili podatke čini nedostupnim za njihove hostove. To je jedan od najjednostavnih oblika mrežnog napada. Umesto pokušavanja krađe ili izmene podataka, cilj ovog napada je sprečavanje pristupa servisu ili resursu. To se obično postiže opterećivanjem servera velikom količinom zahteva, povezujući resurse servera i sprečavanjem ispunjava legitimnih zahteva.

6. IMPLEMENTACIJA WEB APLIKACIJE ZA RAZMENJIVANJE PORUKA POMOĆU EXPRESS-A I SOCKET.IO BIBLIOTEKE

U WebSocket-u, server može da šalje podatke klijentu, ali to takođe može i klijent serveru. Stoga, WebSocket je vrsta komunikacione cevi koja je otvorena u dva smera.
U aplikaciji ćemo pored Socket.io biblioteke koristiti i Express.js web okvir (framework)

6.1 Razvojno okruženje

Prvo što treba uraditi je pokrenuti npm, naš menadžer paketa. Da bismo to učinili, prvo treba da otvorimo Node.Js terminal, kreiramo novi repozitorijum koji će sadržati naš projekat, lociramo se unutar njega i zatim inicijalizujemo npm.

6.2 Arhitektura aplikacije

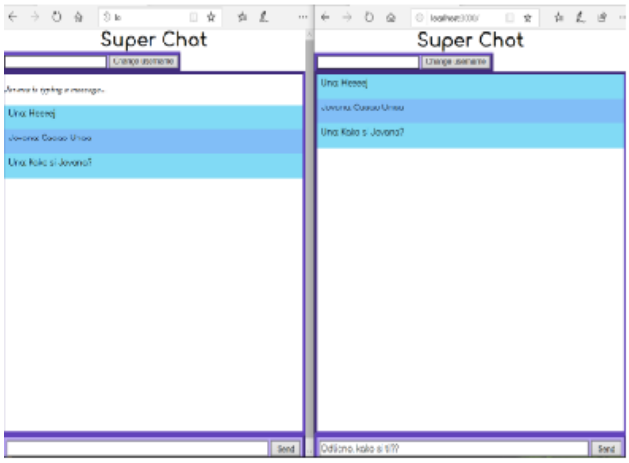
Razlikujemo dva dela u razvoju aplikacije – klijentski i serverski deo. Serverskom stranom će upravljati Node.js, koji će pokretati sve pakete i internet stranice. Klijentski deo će biti prikazan na računaru klijenta. On će imati direktan pristup datotekama (html/css i js).

6.3 Slanje i primanje podataka

Kada se korisnik poveže na aplikaciju, postavićemo mu podrazumevano korisničko ime,npr. „anonimus“. Da bismo to uradili, moramo otići na stranu servera (app.js) i dodati ključ Socketu. U stvari, socket predstavlja svakog klijenta konektovanog na naš server.

Što se tiče razmene poruka, princip je potpuno isti kao i za promenu korisničkog imena.

Za događaj new_message pozivamo io objekat socketa, pozivajući tako sve povezane sockete. Dakle, ova linija koda će zapravo poslati poruku svim socketima, što je nama i cilj jer želimo da poruka koju je korisnik poslao bude vidljiva svima, pa i njemu samom.
Da bismo aplikaciju učinili realnijom, dodali smo feedback poruku koja obaveštava korisnika kada drugi korisnik kuca poruku - porukom “<Korisnik> is typing a message..” To smo postigli dodavanjem JQuery event listener-a na događaj kucanja i šaljemo socket događaj nazvan „typing“. Sa druge strane, slušamo „kucanje“ i emitujemo poruku. Emitovanje znači slanje poruke svima drugima osim socket-u koji ga je pokrenuo.



Slika 1. Konačan izgled aplikacije

7. ZAKLJUČAK

U ovom radu pokazano je da je Node transformisao upotrebljivost JavaScripta, čineći Node kompletnim programskim jezikom. Od internet pretraživača, do serverskih skripti izvan pretraživača, on je omogućio dostupnost runtime okruženja, biblioteke koja sadrži mnoštvo besplatnih korisnih modula koji se mogu uvesti pomoću ugrađenog alata NPM.

Pokazalo se da je postavljanje Node okruženja jednostavno, a dostupan je na svim glavnim operativnim sistemima. Zasniva se na poznatoj sintaksi JavaScripta, ali razlike postoje. Node može biti pomešan sa AJAX-om, ali su dva potpuno različita alata, čija je jedina sličnost to što koriste JavaScript kao bazu.

Pored svih njegovih prednosti, Node ima i neke sigurnosne rupe.

Dakle, ako slučaj upotrebe ne sadrži intenzivne operacije koje opterećuju procesor, niti pristupa bilo kojoj blokadi resursa, mogu se iskoristiti prednosti Node-a i iskusiti brz i skalabilan razvoj aplikacije.

8. LITERATURA

[1] Node.js v5.1.0 Documentation
https://nodejs.org/en/blog/release/v5.1.0/

[2] What is Node?
https://www.amazon.com/dp/B005ISQ7JC/ref=pe_385040_117923520_TE_M1DP

Kratka biografija:

Una Banjanin, rođena je 01.03.1994. u Novom Sadu. Nakon završene Osnovne škole upisuje srednju školu, Gimnaziju „Svetozar Marković" u Novom Sadu. 2013. godine upisuje „Fakultet tehničkih nauka" u Novom Sadu, smer „Računarstvo i automatika". 2018. godine dobija zvanje Diplomirani inženjer elektrotehnike i računarstva. 2018/19. godine upisuje master akademske studije, smer „Primenjene računarske nauke i informatika – inženjering informacionih sistema". kontakt: ubanjanin@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.3
DOI: https://doi.org/10.24867/07BE14Bucko

KARAKTERISTIKE I UPOTREBLJIVOST PROGRAMSKOG JEZIKA GO U MODERNIM APLIKACIJAMA

CHARACTERISTICS AND USABILITY OF THE GO PROGRAMMING LANGUAGE IN MODERN APPLICATIONS

Milica Bučko, Srđan Popov, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu su detaljno istražene i opisane osnovne karakteristike programskog jezika Go počevši od tipova podataka, kreiranja promenljivih preko naredbi toka, paketiranja do konkurentnosti. Razmatrane su prednosti i mane programskog jezika Go u odnosu na programski jezik Java.*

Ključne reči: *Go, Golang, statički tipiziran jezik, Java*

Abstract – *The paper analyzes basic characteristics of the Go programming language starting from data types, creating variables, control flow, packageing till concurencz and parallelism. The advantages and disadvantages of the Go programming language compared to the Java programming language are discussed.*

Keywords: *Go, Golang, static typed language, Java*

1. UVOD

Programski jezik Go, razvila je kompanija Google 2007. Godine u cilju unapređenja produktivnosti u programiranju. Cilj je bio da se jezik Go dizajnira kao statički kompajliran jezik koji se koristi za velike sisteme ali da pri tom programi koji se pišu budu jednostavni za programiranje sa čitljivim kodom, bez prevelikog broja ključnih reči, kao kod dinamički tipiziranih programskih jezika.

Cilj rada je upoznavanje sa programskim jezikom Go, njegovom sintaksom i njegovim karakteristikama. Na studiji slučaja je izvršena komparativna analiza programskog jezika Go sa programskim jezikom Java.

2. PODEŠAVANJE OKRUŽENJA

Procedura instaliranja Go-a na lokalnoj mašini sa Windows operativnim sisemom i podešavanje programskog okruženja putem komandne linije.
Za pokretanje prvog programa pisanog u Go programskom jeziku potrebno je instalirati Go run time i podesiti lokalnu mašinu u cilju pokretanja, buildovanja i izvršavanja programa. Zatim je potrebno uspešno instalirati editor, VSCode. Nakon instalacije VSCode, potrebno ga je konfigurisati kako bi editor znao da rukuje sa kodom.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Srđan Popov, vanr. prof.

2.1. Komande za pokretanje programa

Nakon konfiguracije okruženja moguće je pokrenuti prvi program napisan u Go-u. Neke od osnovnih komandi za pokretanje projekta su go build, go run, go fmt, go install, go get i go test.

3. KARAKTERISTIKE PROGRAMSKOG JEZIKA GO

3.1. Deklaracija i definicija

Kako je Go statički tipiziran jezik prilikom deklaracije promenljive potrebno je dodeliti tip podatka. Pravilo Go programskog jezika je da se svaka deklarisana promenljiva mora iskorititi, mora biti upotrebljena u toku izvršavanja programa. Ovo ne važi samo za promenljive nego i za pakete. Postoji više načina definisanja promenljivih.

3.1.1. Tip promenljive se eksplicitno navodi

Postoje dva slučaja kreiranja promenljivih kada se tip promenljive eksplicitno navodi. Prvi slučaj je definicija i deklaracija promenljive gde se tip promenljive eksplicitno navodi u istom iskazu. Drugi slučaj je deklaracija promenljive gde se tip promenljive eksplicitno navodi a dodjeljivanje se vrši nakon deklaracije u odvojenom iskazu.

3.1.2. Tip promenljive se ne navodi

U slučaju kada se deklaracija promenljive vrši tako što se tip promenljive ne navodi eksplicitno. Na osnovu vrednosti, kompajler treba sam da zaključi koja je vrednost toj promenljivoj dodeljena.

3.1.3. Deklaracije više promenljivih

U slučaju deklaracije više promenljivih, tip promenljive kompajler zaključuje na osnovu vrednosti koja joj se dodeli. Tip promenljive u jednom iskazu ne mora da bude isti. Jedna od dodatnih opcija koje Go jezik nudi, je lak način deklarisanja promenljive bez navođenja ključne reči var, pomoću operatora :=.

Operator **:=** govori kompajleru da je programer naveo ime i vrednost promenljive a on mora sam da zaključi koji tip će joj dodeliti. Ova operacija se koristi samo kada se prvi put navodi nova promenljiva.

3.2. Tipovi podataka

Tipovi podataka koji se mogu koristiti u programskom jeziku Go svrstavaju se u četiri kategorije: osnovni tipovi

(tekstualni, bulean i numerički), složeni tipovi (nizovi i strukture), referentni tipovi (pokazivači, isečci, mape, kanali i funkcije), interfejsni tipovi (interfejsi).

3.2.1. Osnovni tipovi podataka

Osnovni tipovi podataka podeljeni su od tri grupe: numerička grupa (integer, float i kompleksne vrednosti), bool grupa (true i false) i tekstualna grupa predstavlja niz karaktera.

Operatori definisani u Go jeziku su aritmetički operatori, relacioni operatori, logički operatori, bitski operatori i operatori dodele.

3.2.2. Složeni tipovi podataka

Pod složenim tipovima podataka u programskom jeziku Go podrazumevaju se nizovi i strukture.

Nizovi predstavljaju sekvencu elemenata istog tipa, fiksne veličine. Nije moguće kreirati niz takav da sadrži različite tipove podatala (npr. integer i string). Nije moguće menjati veličinu niza nakon što se jednom definiše. Elementima niza pristupa se pomoću indexa, pri čemu prvi element niza ima indeks 0 a poslednji size-1. Pomoću ugrađene funkcije len moguće je dobiti podatak o veličini niza. Još jedna opcija koju Go jezik podržava je poređenje nizova. Nizovi u tom slučaju moraju biti iste dužine i istog tipa podataka.

Strukture su korisnički definisani tipovi koji sadrže nula ili više elemenata istih ili različitih tipova. Strukture se koriste kada korisnik želi da grupiše povezane podatke u jednu celinu. Svaki element strukture mora imati jedinstven naziv, definisan tip i predstavlja jedno polje strukture. Definisanje strukture zahteva navođenje predefinisane reči **type** ispred same definicije strukture. Poljima unutar strukture moguće je pristupiti navođenjem naziva polja.

Go programski jezik podržava ugnježdavanje struktura. Poređenje struktura je takođe podržano ukoliko su dve strukture koje se porede istog tipa.

3.2.3. Referenti tipovi podataka

Pod referentnim tipovima podataka u programskom jeziku Go podrazumavaju se pokazivači, isečci, mape, kanali i funkcije.

Pokazivač je promenljiva koja pokazuje na memorijsku lokaciju neke druge promenljive. Oni čuvaju memorijsku adresu promenljive na koju pokazuju i svesni su gde je locirana kao i vrednosti koja je sačuvana na toj lokaciji.

Isečak ili **slice** u Go programskom jeziku predstavlja jedan deo ili segment niza. Pogled na elemente niza.

Jedna od najkorisnijih stuktura u programiranju jesu heš tabele (hash table), **mape**. U tabeli se čuvaju parovi ključ – vrednost. Ključ mora biti jedinstven.

Funkcije su delovi koda zaduženi za izvršavanje nekog zadatka na osnovu informacija koje su dobile kao ulazni parametar. Kada se jednom kreira, funkcija može da bude pozvana u bilo kom trenutku proizvoljan broj puta. Opciono je da li će funkcija vraća ti neku vrednost ili ne. U Go jeziku, funkcija se definiše pomoću ključne reči **func**.

3.2.4. Interfejsni tipovi

Interfejs predstavlja kolekciju metoda koje određeni objekat može da implementira, što znači da interfejs definiše ponašanje objekta. Ukoliko objekat implementira metode interfejsa kaze se da objekat implementira taj interfejs.

U interfejsu treba da budu navedena imena metoda, ulazni argumenti i povratni tipovi.
Za deklaraciju interfejsa potrebno je navesti **type** alias zajedno za ključnom reči **interface.**

3.2.5. Konstante

Konstante predstavljaju promenljive čija vrednost ne može biti promenjena kad se jednom dodeli. Konstante u Go programskom jeziku se deklarišu pomoću ključne reči const.

3.3. Naredbe kontrole toka

3.3.1. Naredba IF

U Go programskom jeziku, postoje dve opcije upotrebe naredbe if. Korišćenje naredbe if ikoliko je potrebno proveriti neki uslov bez izvršavanja inicijalizacije i korišćenje naredbe if ukoliko je potrebno navesti naredbu koja je znakom ; razdvojena od uslova.

3.3.2. Naredba FOR

U Go programskom jeziku jedina petlja koja postoji je FOR petlja. Prilikom korišćenja FOR petlje za prelazak na sledecu iteraciju koristi se ključna reč **continue** dok se za izlazak iz petlje koristi ključna reč **break**. Takodje, prilikom upotrebe for petlje koristi se i ključna reč **range** koja omogućava iteriranje kroz sekvencijalne strukture kao što su mape, isešci i nozovi.

3.3.3. Naredba SWITCH

Postoji više načina definisanja naredbe switch. Ova naredba omogućava proveravanje vrednosti po slušajevima. Prilikom korisćenja switch naredne koristi se ključna reč **switch** gde se navodi uslov i ključna reč **case** koja proverava različite slučajeve. Switch naredba izvršava prvi case koji zadovoljava zadati uslov. Ukoliko ni jedan case ne zadovoljava uslov izvršiće se **default** izraz.

3.3.4. Naredba DEFER

Naredba defer odlaže izvršavanje navedene funkcije dok se ne završi izvršavanje funkcije u kojoj se nalazi. U trenutku navođenja naredbe defer određuju se argumenti funkcije dok se samo njeno izvršavanje odlaže. U slučaju greške, naredva defer garantuje da će se funkcija izvršiti. Naredba se koristi za operacije koje se obavljaju u paru.

3.4. Upravljanje greškama

Go jezik ne poseduje opciju try/catch metoda za rukovanje greškama već se greške vraćaju kao regularna povratna vrednost.
Greška, predstavlja vrednost koju funkcija može da vrati u slučaju da se tokom izvršavanja desi nešto neočekivano. Ključna reč za grešku je **error** i njena zero vrednost je **nil**.

Greška je zapravo tipa interfejs, dostupna na globalnom nivou i implementira Error metod koji kao povratnu vrednost vraća poruku o greški u String formatu.

3.5. Sakupljanje smeća

U Go programskom jeziku upravljanje memorijom odnosno sakupljanje smeća (garbage collection) obavlja se automatksi. Prvo, sakupljač skenira celu memoriju i pronalazi sve objekte na koje više niko ne referencira, briše ih i time oslobađa memoriju.
Verzijom Go 1.5 uveden je sakupljač smeća kreiran kao konkurentni, trobojni, označi-počisti(mark-sweep) sakupljač. Trobojni sakupljač je sakupljač kod kog je svaki objekat u jednoj od tri boje, bele, sive ili crne i hip segment koji se posmatra kao jedan povezani graf. Hip segment predstavlja deo memorije u kome se dinamički alociraju podaci tokom izvršavanja programa.

3.6. Paketi

Svaki Go program mora biti deo nekog paketa i najjednostavniji program mora imati package main deklaraciju. Ako je program deo main paketa, prilikom izvršavanja naredbe go install biće kreiran binaran (binary) fajl, koji nakon izvršavanja poziva main funkciju programa.
Postoje dva tipa paketa Executable koji generiše fajl koji može da se pokrene i Reusable-code dependensiji i biblioteke, ono što nam pomaže da ponovo koristimo kod u budućim projektima.
Reč main u package main navodi da je u pitanju Executable tip paketa. Ukoliko se izostavi navođenje main-a fajl se neće izvršiti. Kad god je definisan paket main to znači da je definisan paket koji može biti kompajliran i koji se može izvršiti. Takođe, obavezno je i navođenje funkcije main().

3.7. Konkurentnost i paralelizam

Iako se često konkurentnost i paralelizam posmatraju kao jedan pojam, između njih postoje suštinske razlike. Konkurentnost je mogućnost programa da poseduje više niti i da onog trenutka kad je jedna blokirana program izabere neku drugu i krene je izvrršavati. Ovaj proces se dešava na jednom jezgru. Paralelizam je mogućnost programa da poseduje više niti koje se izvršavaju u isto vreme na više različitih jezgara.

3.7.1. Rutine

Svaka nit izvršavanja predstavlja **rutine** (routine) u Go programskom jeziku. Svaka nova rutinu kreira se pomoću ključne reči „go“.

Pošto je Main rutina glavna nit i direktno od nje zavisi dužina izvršavanja programa, onda se zaključuje da onog momenta kad main rutina dođe do kraja svog izvršavanja, program će se završiti. Program nije u stanju da prepozna da postoje još neke nove rutine koje nisu završile svoj posao. Kako bi se izbegao ovaj problem, Go je uveo princip **kanala** (channels).

3.7.2. Kanali

Kanali (channels) se koriste kako bi se omogućila lakša komunikacija i razmena podataka između rutina. Pošto se podaci razmenjuju putem kanala potrebno je obezbediti da tip kanala i tip podataka koji se razmenjuju bude isti.

Kanal se kreira korišćenjem ključne reči **chan.** Kada je kanal napravljen u njega mogu da se šalju podaci i iz njega mogu da se primaju podaci. Kanal je posrednik u komunikaciji između main rutine i child rutina.

4. POREĐENJE PROGRAMSKOG JEZIKA JAVA SA PROGRAMSKIM JEZIKOM GO

4.1. Kontrola pristupa

Java sadrži private, protected, public modifikatore na osnovu kojih se objedinjuje opseg sa različitim nivoima pristupa za podatke, metode i pakete. Go sadrži exported i unexported identifikatore koji su slični public i private modifikatorima u Javi, ali nisu isti, što znači da Go ne sadrži modifikatore.

4.2. Golang nije objektno orijentisan jezik

Pre svega, Go nema objekte već strukture koje donekle mogu simulirati ponašanje objekata pa odmah u startu nema jednu od najvažnijih osobina objektno orijentisanog jezika, a to je predstavljanje entiteta putem objekata. Još jedna bitna razlika u odnosu na Javu je ta što Go jezik nema nasleđivanje zbog toga što ne podržava tradicionalni polimorfizam koji se ostvaruje putem nasleđivanja. Umesto nasleđivanja, U Go-u se koristi kompozicija.

4.3. Redosled deklaracije

Još jedna od velikih razlika između ova dva programska jezika je redosled definisanja (deklarisanje) varijabli.

Redosled navođenja može biti zbunjujući jer se razlikuje od velikog broja programskih jezika, ali suštinski se ništa ne gubi jer dobijamo isti efekat deklaracije kao i u bilo kom programskom jeziku.

4.4. Pokazivači

Java ne podržava pokazivače jer su kreatori želeli da dobiju na jednostavnosti i da spreče korisnika da direktno upravlja memorijom, te su uveli garbage collector koji se sam brine za oslobađanje zauzete memorije objekta koji više nisu u upotrebi.

Sa druge strane, Go podrazumeva pokazivače koji generalno imaju par svojih prednosti kao što su pristupanje putem adrese promenljive, dinamička alokacija memorije, ubrzavanje izvršavanja programa itd.

4.5. Funkcija kao argument

U Go programskom jeziku, funkcija može da se koristi kao varijabla koja će biti prosleđena nekoj drugoj funkciji.
Počevši od Java 1.8 i korišćenja lambda izraza, dobijena je slična funkcionalnost ali objekti koji se prosleđuju u drugim funkcijama nisu zabravo funkcije nego one-function objekti.

4.6. Funkcije mogu vratiti više argumenata

Jedna od prednosti Go jezika u odnosu na Javu, je ta što je u Go-u podržano da funkcije mogu vraćati više argumenata. Ovu karakteristiku je pre svega omogućila upotreba pokazivača.

4.7. Generics

Generics doprinosi kompleksnosti programa, mnogo konverzija i posla u runtime-u. Iz tog razloga Go ne podržava generics. Shodno tome, u Go jeziku je potrebno mnogo više koda da bi se predstavilo isto ponašanje za različite tipove podataka, nego što je to slučaj sa Javom.

4.8. Anotacije

Za razliku od Java jezika, Go jezik ne podržava anotacije. Ovo je jedna od loših strana Go programskog jezika. Anotacije u Javi su jednostavan i čitljiv način da prikažemo ponašanje za određene delove koda. Ovo direktno zači da svu logiku koje anotacije nose sa sobom u Go jeziku moramo da pišemo ručno, što zahteva dosta više posla i donosi mnogo više problema pri održavanju.

4.9. The profiler

Profiler alatke u Go-u su brže i jednostavnije za korišćenje. Pomažu korisniku da vidi na koji način je alocirana memorija i koliki je stepen iskorišćenosti CPU-a za sve delove programa. To se može ilustrovati putem grafa, pa samim tim je ekstremno olakšan posao ukoliko neki deo programa treba da se optimizuje. Java takođe sadrži profilere, kao što je Java Visual VM ali nisu jednostavni kao Go prifileri.

4.10. Overloading

Overloading nije podržan u programskom jeziku Go. Go zahteva da sve funkcije imaju unikatna imena. Praktično, u Javi možemo imati dve funkcije koje imaju isto ime, a različite tipove parametara i to će biti sasvim ispravno.

4.11. Ostalo

Još jedna od dobrih karakteristika je velika količina informacija koje korisnik može da dobije o ovom jeziku. Go od samog početka ima odličnu dokumentaciju i odlične preporuke na koji način je najbolje koristiti Go i kakav stil koristiti u kojim situacijama.

Go jezik ima veoma jednostavnu sintaksu, za razliku od Java jezika, korisnik ne mora da navodi ; da bi naglasio završetak linije. Sintaksa Go jezika se sastoji iz veoma malog broja ključnih reči.

Za razliku od Java jezika, Go ima ugradjenu podršku za kompleksne brojeve koja je jednostavna i intuitivna.

5. ZAKLJUČAK

Programski jezik Go je relativno nov jezik ali je zahvaljujući svojim dobrim karakteristikama uspeo da se zadrži u svetu programiranja. Jezik se konstantno unapređuje i razvija i svakom novom verzijom uvodi znatna poboljšanja u performansama. Jedna od najvećih prednosti Go jezika jeste brzina kompajliranja koja je mnogo bolja u odnosu na druge jezike.

Google je kreirao ogroman broj zvaničnih tutorijala i omogućio da Golang bude open-source tako da trenutno postoji ogromna količina dodataka i paketa. Ukoliko se radi sa ugradnim (embeded) sistemima koji raspolažu sa ograničenom memorijom onda Go nije najbolja opcija koja se može izabrati za implementaciju. Golang je odličan izbor za web aplikacije koje su danas sastavljene od velikog broja spoljnih servisa i baza podataka. To su aplikacije koje su najčešće organizovane pomoću arhitekture mikroservisa.

Popularnost Go jezika raste i smatra se da će Go jezik u narednih nekoliko godina biti svrstan u najpopularnije jezike za serverske aplikacije.

6. LITERATURA

[1] www.udemy.com/go-the-complete-developers-guide/
[2] medium.com/rungo/

Kratka biografija:

Milica Bučko, je rođena 02.09.1994. godine u Novom Sadu. Osnovnu školu „Vuk Karadžić“, u Bačkoj Palanci, završila je 2009. godine. Gimnaziju „20. oktobar“ u Bačkoj Palanci, završila je 2013. godine. Iste te godine upisala je Fakultet tehničkih nauka u Novom Sadu, smer Računarstvo i automatika. 2017. godine dobija zvanje Diplomirani inženjer elektrotehnike i računarstva. 2017/18. godine upisuje master akademske studije, smer „Primenjene računarske nauke i informatika – Elektronsko poslovanje”.
kontakt: milica.milica.b@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/07BE16Mrkaljevic

PODEŠENJE RELEJNE ZAŠTITE ENERGETSKIH TRANSFORMATORA UNUTAR TRANSFORMATORSKE STANICE 220/110/35/6 kV

ADJUSTMENT OF RELAY PROTECTION OF POWER TRANSFORMERS WITHIN 220/110/35/6 kV TRANSFORMER STATION

Vojislav Mrkaljević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je opisano podešenje relejne zaštite energetskih transformatora unutar transformatorske stanice TS 220/110/35/6 kV. Opisane su TEHNIČKE PREPORUKE, TEHNIČKO UPUSTVO, kao i sam primer podešenja relejne zaštite.*

Ključne reči: *Transformatorska stanica TS 220/110/35/ 6kV, Tehničke preporuke, Tehničko upustvo*

Abstract – *According to the title, the paper describes the setting of relay protection of power transformers within the transformer station TS 220/110/35/6 kV. TECHNICAL RECOMMENDATIONS, TECHNICAL INSTRUCTION, as well as an example of setting of relay protection are described.*

Keywords: *Transformer station TS 220/110/35 / 6kV, Technical recommendations, Technical manual*

1. UVOD

Tema rada jeste podešavanje relejne zaštite transformatorske stanice TS 220/110/35/6 kV. Rad se sastoji iz dva dela, teorijskog i praktičnog. Unutar praktičnog dela dat je primer podešenja relejne zaštite TS 220/110/35/6 kV.

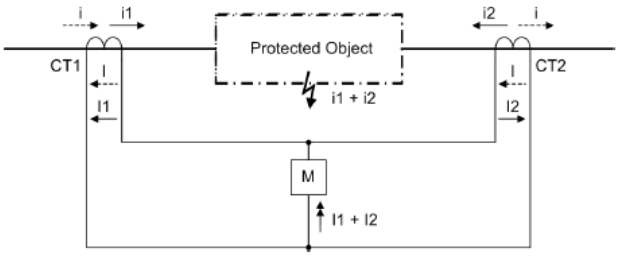
Kvarovi u elektroenergetskom sistemu su relativno česta pojava. Rizik od kvarova je veliki zbog same prirode elektroenergetskog sistema i vanjskih uslova. Gotovo su svi kvarovi razorne moći čime predstavljaju veliku opasnost po čovjeka i opremu. Nastankom kvara potrošači nerijetko ostaju bez napajanja što za njih predstavlja materijalne gubitke.

Zbog karaktera kvarova i njihove učestalosti zaštita elektroenergetskog sistema je imperativ kako u prijenosu tako i u distribuciji. Zadatak zaštite jeste sprečavanje nastanka kvarova i što brže uklanjanje kada do kvara dođe. Razvojem tehnologije pojavile su se mnoge vrste zaštite.

2. ZAŠTITA ENERGETSKIH TRANSFORMATORA

Kao osnovna zaštita svakog energetskih transformatora odabrana je diferencijalna zaštita. Diferencijalna zaštita štiti transformator od kvarova unutar štićene zone.

Štićena zona podrazumijeva prostor između strujnih transformatora na svakoj strani energetskog transformatora, na koje se zaštita priključuje. U normalnom pogonu transformatora, kao i u slučaju kvarova izvan štićene zone, diferencijalna zaštita mora ostati neosjetljiva. Na slici 1 prikazan je primer diferencijalne zaštite dvo-namotajnog transformatora:

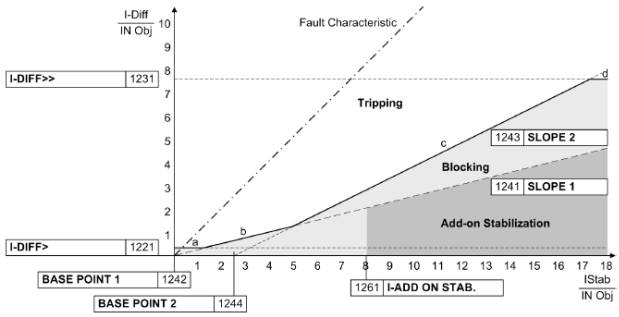


Slika 1. Primjer diferencijalne zaštite dvonamotajnog transformatora

2.2. DIFERENCIJALNA ZAŠTITA

Usled snažnih kratkih spojeva izvan zone štićenja praktično se može desiti da različiti strujni transformatori na suprotnim stranama energetskog transformatora imaju različite prekostrujne faktore i različite amplitudne i ugaone greške, pa može doći do pojave diferencijalne struje i neželjene odrade zaštite. To je dakle sprečeno odabirom zaštite sa stabilizovanom proradnom karakteristikom.

Stabilizovana proradna karakteristika diferencijalne zaštite transformatora prikazana je na slici 2.



Slika 2. Stabilizovana proradna karakteristika diferencijlane zaštite transformatora

Diferencijalna struja određena je jednačinom [1]:

$$I_{DIFF} = |\overrightarrow{I_1} + \overrightarrow{I_2}| \quad (1)$$

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dejan Jerkan, docent.

Stabilizaciona struja određena je jednačinom [1]:

$$I_{STAB} = |\overrightarrow{I_1}| + |\overrightarrow{I_2}| \quad (2)$$

U normalnom pogonu i u slučaju vanjskog kvara struja I_1 ulazi u štićeni objekat, a struja I_2 izlazi iz štićenog objekta, a iste su po veličini. Tada važi da je $\overrightarrow{I_1} = \overrightarrow{I_2}$, i $|\overrightarrow{I_1}| = |\overrightarrow{I_2}|$.

Iz toga sledi da je [1]:

$$I_{DIFF} = 0, I_{STAB} = 2|\overrightarrow{I_1}| \quad (3)$$

odnosno nema prorade diferencijalne zaštite.
U slučaju unutrašnjeg kvara, koji se napaja sa obje strane, obje struje teku prema transformatoru. Ako npr. pretpostavimo da je $|\overrightarrow{I_1}| = |\overrightarrow{I_2}|$, tada je [1]:

$$I_{DIFF} = I_{STAB} = 2|\overrightarrow{I_1}| \quad (4)$$

U ovom slučaju dolazi do sigurne odrade diferencijalne zaštite, kada su diferencijalna i stabilizacijona struja jednake.

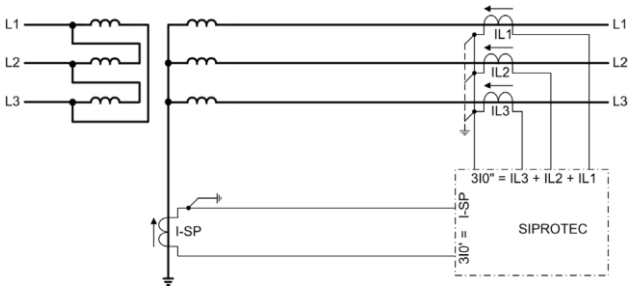
U slučaju da je kvar napojen samo sa jedne strane (npr. sa primarne), tada pretpostavimo da je $|\overrightarrow{I_2}| = 0$ i tada je [1]:

$$I_{DIFF} = I_{STAB} = |\overrightarrow{I_1}| \quad (5)$$

I u ovom slučaju dolazi do sigurne odrade diferencijalne zaštite, kada su diferencijalna i stabilizacijona struja jednake.

2.3. OGRANIČENA ZEMLJOSPOJNA ZAŠTITA TRANSFORMATORA

Na slijedećoj slici prikazan je način priključka ograničene zemljospojne zaštite.



Slika 3. Priključak ograničene zemljospojne zaštite transformatora

Ova zaštita se aktivira takođe u uređaju kao dopunska funkcija diferencijalnoj zaštiti transformatora, i ima ulogu osjetljive zaštite u slučaju zemljospoja unutar zone štićenja preko velike impedanse, koju diferencijalna zaštita može da ne osjeti.

U slučaju kvara sa zemljom izvan štićene zone, razlika (diferencija) struja u faznim vodičima (3Io) i struje kroz zvjezdište transformatora je jednaka 0 A. Naravno, isto kao i kod diferencijalne zaštite u slučaju snažnih kvarova izvan štićene zone uslijed različitih karakteristika strujnih transformatora u faznim vodičima i strujnog transformatora u zvjezdištu, može se pojaviti diferencijalna struja, dovoljna za odradu zaštite. Iz tog razloga, ograničena zemljospojna zaštita takođe ima stabiliziranu proradnu karakteristiku.

U slučaju kvara unutar štićene zone pojavljuje se razlika između struja u faznim vodičima (3Io) i struje u zvjezdištu transformatora i ograničena zemljospojna zaštita odrađuje trenutno.

2.4. ZAŠTITA OD PREOPTEREĆENJA

Iako je, kako je već rečeno, osnovna zaštita transformatora diferencijalna zaštita, u zaštitnim uređajima se aktiviraju i zaštita od preopterećenja (prekostrujna zaštita) i kratkospojna zaštita.

Zaštita od preopterećenja ima ulogu da zaštiti transformator od situacija kada nema kvara, nego je transformator preopterećen, a takođe predstavlja i krajnju rezervnu zaštitu u slučaju zatajenja ostalih.

S obzirom da se energetski transformator element koji se može preopteretiti neko vrijeme, za zaštitu od preopterećenja odabrana je normalno inverzna karakteristika (strujno-ovisna karakteristika), kod koje se vrijeme do odrade mijenja u zavisnosti od struje opterećenja prema slijedećoj zakonitosti [3]:

$$t = \frac{0{,}14}{\left(\frac{I}{Ip}\right)^{0{,}02} - 1} Tp(\mathrm{s}) \quad (6)$$

gdje je:

- t – vrijeme do odrade zaštite;
- I – struja opterećenja;
- Ip – podešena vrijednost struje;
- Tp – podešena vremenska konstanta.

Podešena vrijednost Ip odabrana je za svaki transformator u skladu sa njegovom nominalnom strujom, ili u skladu sa nominalnom strujom strujnog transformatora na koji se priključuje. Vremenska konstanta Tp odabrana je tako da se zadovolji uslov selektivnosti zaštita.

2.5. PRIMER DELOVANJA RELEJNE ZAŠTITE U SLUČAJU KVARA

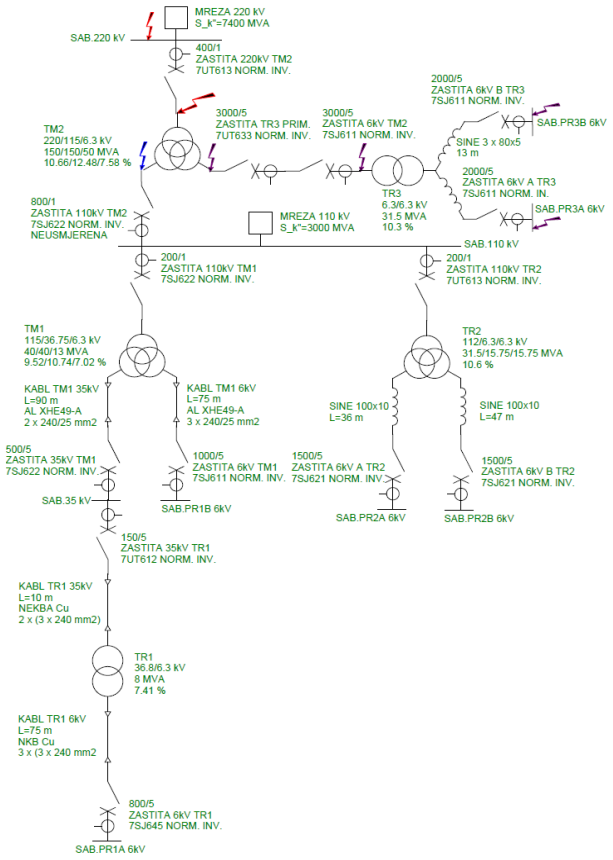
Na slici 4. prikazana je jednopolna šema trafostanice 220/110/35/6 kV kao i specifične tačke kvarova (dvopolni i tropolni) [2].

U slučaju kvara na 6 kV strani transformatora TM2 djelovaće sigurno diferencijalna zaštita, isključiti sve tri strane TM2 i na taj način odvojiti mjesto kvara od ostatka sistema za veoma kratko vrijeme.

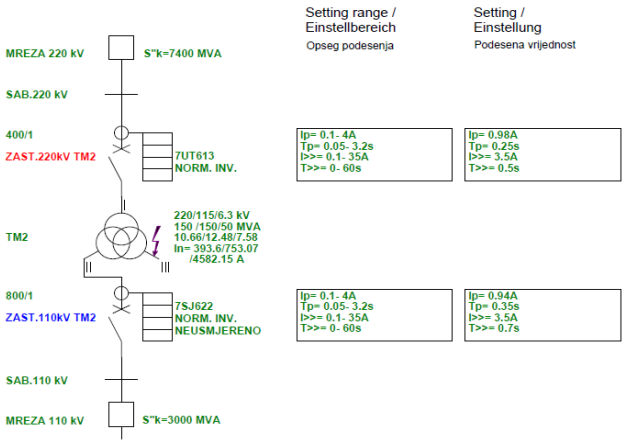
Ipak, u slučaju nereagovanja diferencijalne zaštite vidljivo je sa slike 5. i slike 6. da će struja kvara na 220 kV strani biti dosta mala (reda 250 A) i nedovoljna za djelovanje kratkospojne zaštite, pa čak i zaštite od preopterećenja. Na 110 kV strani je situacija nešto bolja, struja je reda 2200 A i za očekivati je djelovanje zaštite od preopterećenja. U svakom slučaju, ovaj kvar je dosta

nepovoljan u slučaju nereagovanja diferencijalne zaštite, pa treba svakakao izvesti djelovanje kratkospojne zaštite 110 kV strane na prekidač 220 kV.

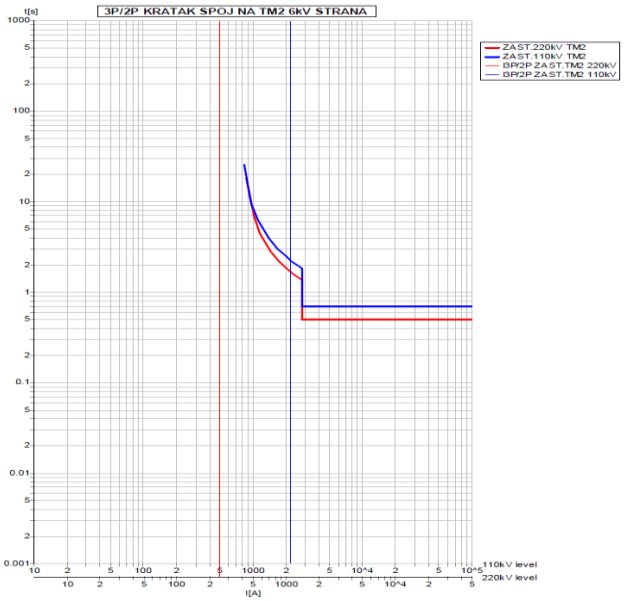
Ipak, praktično nema razloga da dođe do nereagovanja diferencijalne zaštite u slučaju ovakvog kvara.



Slika 4. jednopolna šema trafostanice 220/110/35/6 kV



Slika 5. simulacija kvara na 6 kV strani transformatora TM2



Slika 6. odziv relejne zaštite

3. ZAKLJUČAK

U ovom radu opisane su zaštite dalekovoda, kao i zaštite energetskih transformatora. Akcenat je bio na zaštitama energetskih transformatora, kao ključnih elemenata unutar transformatorske stanice.

Prikazane su simulacije kratkih spojeva u najkritičnijim tačkama energetskog postrojenja TS 220/110/35/6 kV i njihov odziv, selektivnost i funkcionalnost.

Najvažnija (osnovna) zaštita energetskih transformatora jeste diferencijalna zaštita.

4. LITERATURA

[1] Tehničko uputstvo za podešavanje zaštita energetskih transformatora, elektroistok tu-za-03/2.

[2] Studija podešenja relejne zaštite, Cet Energy.

[3] Tehnička preporuka broj 4 - Opšti deo (TP 4 Opšti deo) – Primena zaštite i lokalne automatike u distributivnim mrežama 10 kV, 20 kV, 35 kV i 110 kV.

Kratka biografija:

Vojislav Mrkaljević rođen je u Hrgama, opština Zavidovići 18.04.1994. godine. Master rad je odbranio 2019 godine na Fakultetu tehničkih nauka, Elektrotehnika i računarstvo - Energetska elektronika i električne mašine.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/07BE17Trakilovic

IMPEDANTNA METODA ZA ODREĐIVANJE LOKACIJE JEDNOPOLNOG KVARA U DISTRIBUTIVNIM MREŽAMA

IMPEDANCE-BASED METHOD FOR LOCATING SINGLE LINE-TO-GROUND FAULT IN DISTRIBUTION NETWORKS

Slavka Trakilović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je razmatrana lokacija kvara kao jedna od najznačajnijih komponenti upravljanja kvarom u distributivnim mrežama. Detaljno su objašnjene impedantne metode za jednopolni kvar kao najčešći i najkomplikovaniji za lociranje. Takođe je obrađen uticaj uzemljenja zvjezdišta na proračun struja kvara, a samim tim i na lokaciju kvara. Svi proračuni su realizovani pomoću programa koji je za potrebe ovog rada razvijen u programskom jeziku Fortran. Dobijeni rezultati su predstavljeni tabelarno i kroz dijagrame i izvršena je njihova analiza.*

Ključne reči: *Distributivna mreža, Impedantna metoda, Lokacija kvara*

Abstract – *This work considers fault location as one of the most important components of fault management in distribution networks. It gives detailed explanation of the impedance-based methods for the single line-to-ground fault as the most frequent and most complicated for determining the fault site location. Besides, it considers the influence of the neutral grounding on short-circuit calculation, and, by the same token, on the fault location. All calculations are done using a program specially designed for the need of this work, developed in the programming language Fortran. The obtained results are presented in the form of tables and diagrams, and they were appropriately analyzed.*

Keywords: *Distribution network, Impedance-based method, Fault location*

1. UVOD

Glavni cilj svakog preduzeća za distribuciju električne energije je da održi što kvalitetnije snabdijevanje potrošača električnom energijom uz što manje troškova. Moderne distributivne mreže (DM) su ogromni sistemi sa velikim brojem elemenata, i kao posljedica toga događaju se različiti kvarovi.

Ti kvarovi mogu biti uzrokovani atmosferskim uticajima, uslovima infrastrukture, konfigu-racijom terena. Pri pojavi kvara, nakon djelovanja zaštite, najvažnije je što prije popraviti kvar i obnoviti napajanje potrošača. Prvi korak u ovom procesu je određivanje lokacije kvara.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je dr Duško Bekut, red. prof.

Postupak lokacije kvara sastoji se iz dva koraka. U prvom koraku se procjenjuje gdje se nalazi mjesto sa kvarom, dok se u drugom koraku nalazi stvarno mjesto kvara, pri čemu se polazi od procijenjenog mjesta kvara [1].

Prva pretpostavka pri razmatranju lokacije kvara jeste da je DM opremljena brzim mjernim jedinicama koje povezivanjem sa stabilnim računarskim sistemom omogućuju brz proračun kvarova. Uz brze mjerne jedinice postoje razna tehnička izvođenja koja omogućavaju pouzdanu lokaciju kvarova na mreži.

Drugi dio rada je posvećen značaju i osnovnim principima lociranja kvarova, a treći prilikama u mreži za vrijeme trajanja jednopolnog kvara. Impedantna metoda se razmatra u četvrtom dijelu, a peti dio verifikaciji metode za lokaciju kvara. U poslednja dva dijela su dati zaključak i literatura.

2. ZNAČAJ I OSNOVNI PRINCIPI LOCIRANJA KVAROVA

Poznavanje tačne lokacije kvara, bez obzira da li je kvar prolazan ili trajan, je bitno za unapređenje rada distributivnog sistema, odnosno povećanja pouzdanosti napajanja potrošača i smanjenja troškova rada sistema. Lociranje kvara je proces procjene mjesta kvara sa što većom tačnošću. Načini implementacije funkcija za lociranje kvara su [2]:

- mikroprocesorskim zaštitnim relejima,
- digitalnim snimačima kvara,
- samostalnim lokatorima kvara,
- programima za analizu sistema poslije kvara.

Funkcija za lociranje kvara se uglavnom u praksi implementira u mikroprocesorske releje kao dodatna funkcija. Digitalni snimači kvara omogućavaju jedno-stavnu i jeftinu implementaciju ovakvih funkcija. Samostalni lokatori koriste sofisticirane algoritme i primjenjuju se kada su prihvatljivi veći troškovi. Programi za analizu sistema poslije kvara sadrže u sebi algoritme za procjenu mjesta kvara i uglavnom se koriste za upravljanje relejima [2].

Većina algoritama za procjenu mjesta kvara je zasnovana na principu mjerenja impedanse korišćenjem napona i struja osnovne frekvencije. Ulazni signali se mogu, u zavisnosti od mogućnosti lokatora kvara, klasifikovati na ulazne signale koji se dovode samo sa jednog kraja voda i ulazne signale koji se dovode lokatoru sa oba kraja voda [3]. Faktori koji utiču na tačnost lociranja kvara su:

- mjesto kvara,
- tip kvara,

- nehomogenost vodova,
- netačnost procjene parametara nadzemnih vodova i kablova,
- prisustvo otočnih elemenata, kao što su kondezatorske baterije,
- pogrešna identifikacija tipa kvara,
- početni fazni stav,
- tačnost analogno/digitalnih konvertora, itd.

Način uzemljenja neutralne tačke distributivnih transformatora utiče na procjenu mjesta jednopolnog kvara u DM. Uticaj načina uzemljenja neutralne tačke na lociranje mjesta kvara je detaljnije objašnjen u narednim glavama.

3. PRILIKE U MREŽI TOKOM JEDNOPOLNOG KVARA

Statistički podaci pokazuju da se najčešće u DM pojavljuju jednopolni dozemni kvarovi, koji u zavisnosti od uzemljenja zvjezdišta mogu biti zemljospoj ili jednopolni kvar. Iz navedenog razloga, u nastavku rada je posvećena pažnja prilikama u mreži za vrijeme ovog kvara, prenaponima do kojih dolazi, naponu dodira i koraka, specifičnoj otpornosti tla [4].

3.1 Prilike u mreži za vrijeme zemljospoja

Prema statistici kvarova, najčešći su jednopolni dozemni – čine 50% od ukupnog broja kvarova. U zavisnosti od uzemljenja neutralne tačke, dozemni kvar se tretira kao zemljospoj ili jednopolni kvar. Jednopolni kvar sa zemljom ima značenje kvara ako je zvjezdište mreže direktno uzemljeno [5]. U tom slučaju iznos veličine struje jednopolnog kvara reda je veličine iznosa struje tropolnog kvara. U mrežama sa izolovanim zvjezdištem, jednopolni kvar se tretira kao zemljospoj. Struje zemljospoja su kapacitivne i teku kroz dozemne kapacitete.

U mrežama koje su direktno uzemljene, tokom jednopolnog kvara struja kvara teče samo u fazi sa kvarom i vraća se kroz zemlju. Duž zdravih faznih vodova fazni naponi su konstantni, dok na faznom vodu koji je obuhvaćen kvarom napon opada od nazivnog napona u izvoru do nule u tački kvara [5].

U mrežama sa izolovanim zvjezdištem, tokom zemljospoja, napon faze pod kvarom u tački zemljospoja jednak je nuli, dok fazni naponi zdravih faza postaju jednaki linijskim naponima. Kroz dozemni kapacitet faze u kvaru ne teče struja, jer je fazni napon faze u kvaru nula. Posljedica zemljospoja je termičko opterećenje vodova i izolatora usljed električnog luka na mjestu kvara. Električni luk se može proširiti i na zdrave faze i na taj način se pretvoriti u kvar. Struja zemljospoja nije velika, ali na mjestu gdje utiče u zemlju stvara velike i po život opasne potencijalne razlike. Zbog povećanja napona zdravih faza, povećava se i mogućnost pojave korone.

Kapacitivne struje pri jednopolnom kvaru ne teku samo kroz vod u kvaru, nego i u zdravim vodovima. Smjer im je prema sabirnicama, a veličina proporcionalna kapacitetu zdrave dionice. Kablovi najvećim dijelom utiču na veličinu kapacitivne komponente struje zemljospoja, dok nadzemni vodovi veoma malo utiču. Visina kapacitivne komponente struje zemljospoja galvanski povezane mreže je osnovni kriterijum za odluku o prelasku mreže na rad sa uzemljenim zvjezdištem.

4. IMPEDANTNA METODA ZA LOKACIJU KVARA

Postoje različiti algoritmi za lociranje kvara, koji su bazirani na mjerenju impedanse kvara. Tokom implementacije tehnika zasnovanih na mjerenju impedanse javljaju se problemi zbog specifičnosti DM, poput nehomogenih vodova sa priključenim opterećenjima i otcjepima po cijeloj dužini. Generalno, svi impedantni algoritmi sastoje se iz tri koraka:

- Na osnovu režima prije kvara određuje se ekvivalentna impedansa direktnog i nultog redoslijeda uvažavajući topologiju, priključena opterećenja, kao i parametre mreže.
- Nakon detektovanja kvara računa se ekvivalentna impedansa petlje kvara koja zavisi od tipa kvara (međufazni kvarovi, kvarovi sa zemljom). Na osnovu izračunate impedanse je moguće doći do lokacije kvara.
- U slučaju kada postoji više potencijalnih mjesta kvara, potrebno je primijeniti adekvatni postupak da bi se pronašlo najvjerovatnije rješenje.

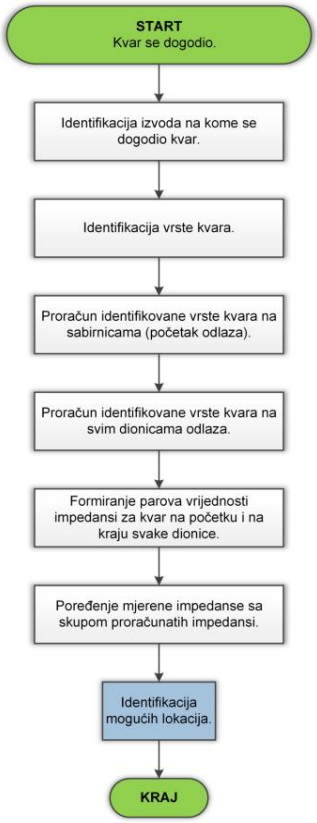
Metoda pogonske impedanse se zasniva na principima koji se primjenjuju kod distantne zaštite. Sam distantni relej nije dovoljan za lociranje kvara. On se podešava u nekoliko stepeni, gdje se za svaki od njih podešava odgovarajuća impedansa u okviru koje relej treba da odreaguje.

Na osnovu vrijednosti struje i napona koji su zabilježeni mjernim uređajem u SN polju napojnog transformatora VN/SN, računa se impedansa između mjerne jedinice i mjesta kvara. Na osnovu prorade releja se zna na kom fideru se desio kvar, a na osnovu mjerenja se dobija i tip kvara, tako što se identifikuju faze u kojima je povećana struja (odnosno, snižen napon). Kao i kod svih distantnih releja, za mjeru udaljenosti uzima se pogonska impedansa voda do mjesta kvara i ta impedansa odgovara impedansi za direktni režim. Relej je podešen da mjeri impedansu direktnog redoslijeda, tačnije dovode mu se naponi i struje iz čijeg će se količnika dobiti impedansa direktnog redoslijeda. Skup primarnih dionica sa kvarom se identifikuje pomoću sljedeće relacije:

$$Im\{Z_{ff1}\} \leq Im\{Z_{ff}\} \leq Im\{Z_{ff2}\} \tag{4.1}$$

gdje su Z_{ff1} i Z_{ff2} impedanse dionice na početku i kraju, a sa Z_{ff} je označena izračunata impedansa.

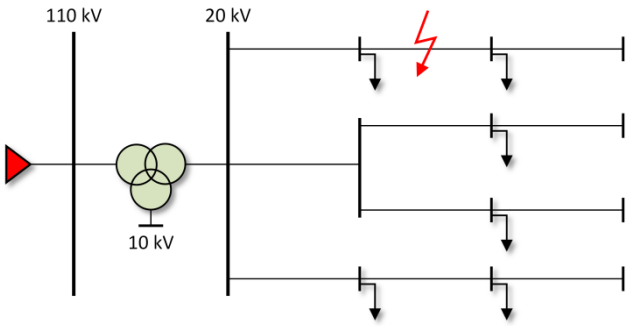
Kako bi se odredio skup sekundarnih lokacija kvara uzimaju se u obzir greške u mjerenju, i smatra se da one iznose oko 2%. Iz tog razloga se izračunata impedansa Z_{ff} množi sa 0.98, odnosno 1.02 i sada se ta nova vrijednost poredi sa Z_{ff1} i Z_{ff2}. Ovaj algoritam daje najbolje rezultate za velike struje kvara pa se on u mrežama u kojima su struje kvara male ne primjenjuje. Pogodan je za korišćenje i u kablovskim i u vazdušnim mrežama. Njegov nedostatak je mala preciznost pri procjeni mjesta kvara na početnim dionicama izvoda. Javlja se i problem kod primjene na mješovitim izvodima gdje se koeficijent zemljospoja mijenja u zavisnosti od dužina kablovskih i vazdušnih dionica od početka izvoda do mjesta kvara. Globalni dijagram impedantnog metoda dat je na slici 4.1.



Slika 4.1 – Blok dijagram impedantnog algoritma lokacije kvara

5. VERIFIKACIJA METODE ZA LOKACIJU KVARA

Nakon opisa impedantnog algoritma, u nastavku rada su dati primjeri procjene lokacije mjesta kvara impedantnim algoritmom. Za realizaciju primjera korišćena je distributivna test mreža prikazana na slici 5.1. Obrađeni su slučajevi lokacije kvara impedantnom metodom u zavisnosti od načina uzemljenja ekvivalentiranog tronamotajnog transformatora. Svi proračuni koji su od interesa izvršeni su pomoću programskog jezika Fortran.



Slika 5.1 – Distributivna test mreža

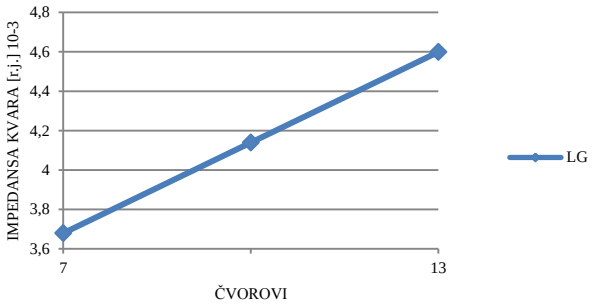
Obrađeni su slučajevi lokacije kvara impedantnom metodom u zavisnosti od načina uzemljenja zvjezdišta ekvivalentiranog transformatora. Razmatrano je nekoliko slučajeva: kada je zvjezdište sekundara uzemljeno preko male otpornosti, zatim kada je zvjezdište sekundara uzemljeno direktno i kada je zvjezdište sekundara izolovano. Nakon proračuna lokacije kvara na distributivnoj test mreži, analizirani su i prikazani dobijeni rezultati, tabelarno i grafički.

Prvi primjer – simuliran je jednopolni kvar na sredini druge dionice izvoda (4007) i impedantnim algoritmom je urađena lokacija kvara za direktno uzemljenu mrežu. Rezultati proračuna lokacije jednopolnog kvara impedantnim algoritmom u relativnim jedinicama, za direktno uzemljenu mrežu, su predstavljeni u tabeli 5.1.

Tabela 5.1 – Rezultati proračuna lokacije jednopolnog kvara impedantnim algoritmom za direktno uzemljenu mrežu

Izmjerena impedansa kvara	Sekcija sa kvarom		
	Impedansa čvor 1	Impedansa čvor 2	ID sekcije
0.00413940	0.00367948	0.00459929	4007

Na slici 5.2 grafički su prikazani dobijeni rezultati (označeni rombovima) proračuna lokacije kvara impedantnim algoritmom, pri simulaciji jednopolnog kvara, u slučaju direktno uzemljene test mreže. Za ovaj kvar imaginarni dio impedanse je 0.00413940 [r.j.] što je veće od imaginarnog dijela impedanse za čvor na početku druge dionice izvoda (dionica 4007 u mreži), dok je istovremeno manji od dijela impedanse za čvor na kraju druge dionice izvoda (dionica 4007 u mreži). Time je ispunjen uslov (4.1), što znači da se kvar nalazi na dionici 4007. Preciznija kriva promjene vrijednosti impedanse duž izvoda dobila bi se kada bi se na više mjesta simulirali kvarovi, pri čemu bi se za interpolaciju između tačaka, umjesto pravih, koristio polinom višeg stepena (drugog, trećeg). Dijagram na slici 5.2 se može iskoristiti da se iz presjeka interpolacije krive i mjerene vrijednosti odredi orijentaciona mikro lokacija kvara.



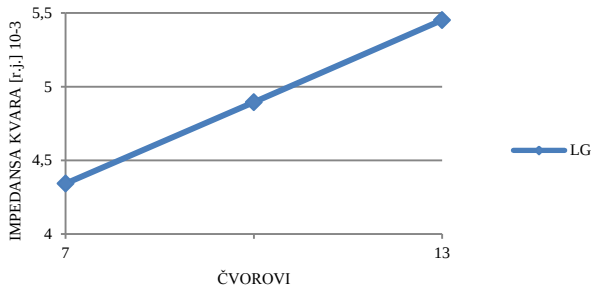
Slika 5.2 – Grafički prikaz rezultata impedantnog algoritma za direktno uzemljenu mrežu

Drugi primjer – simuliran je jednopolni kvar na sredini druge dionice izvoda i impedantnim algoritmom je urađena lokacija kvara za uzemljenu mrežu preko male otpornosti Ruzem. = 40 [Ω]. Rezultati proračuna lokacije jednopolnog kvara impedantnim algoritmom u relativnim jedinicama, za uzemljenu mrežu preko male otpornosti, su predstavljeni u tabeli 5.2.

Tabela 5.2 – Rezultati proračuna lokacije jednopolnog kvara impedantnim algoritmom za mrežu uzemljenu niskoomskim otpornikom

Izmjerena impedansa kvara	Sekcija sa kvarom		
	Impedansa čvor 1	Impedansa čvor 2	ID sekcije
0.00489498	0.00434250	0.00545224	4007

Na slici 5.3 grafički su prikazani dobijeni rezultati proračuna lokacije kvara impedantnim algoritmom, pri simulaciji jednopolnog kvara, u slučaju uzemljene mreže preko niskoomskog otpornika. Za ovaj kvar imaginarni dio impedanse je 0.00489498 [r.j.] što je veće od imaginarnog dijela impedanse za čvor na početku druge dionice izvoda, dok je istovremeno manji od dijela impedanse za čvor na kraju druge dionice izvoda. Time je ispunjen uslov (4.1), što znači da je kvar na dionici 4007. Kao rezultat procjene mjesta kvara dobijena je upravo dionica na kojoj je simulirano mjesto kvara.



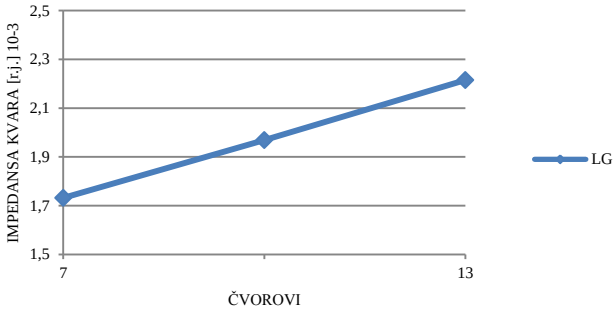
Slika 5.3 – Grafički prikaz rezultata impedantnog algoritma za mrežu uzemljenu niskoomskim otpornikom

Treći primjer – simuliran je jednopolni kvar na sredini druge dionice izvoda i impedantnim algoritmom je urađena lokacija kvara za mrežu sa izolovanim zvjezdištem. Rezultati proračuna lokacije jednopolnog kvara impedantnim algoritmom za mrežu sa izolovanim zvjezdištem su predstavljeni u tabeli 5.3.

Tabela 5.3 – Rezultati proračuna lokacije jednopolnog kvara impedantnim algoritmom za mrežu sa izolovanim zvjezdištem

Izmjerena impedansa kvara	Sekcija sa kvarom		
	Impedansa čvor 1	Impedansa čvor 2	ID sekcije
0.01968451	0.01731880	0.02214944	4007

Jedna od osnovnih prednosti mreže sa izolovanom neutralnom tačkom jeste mogućnost samogašenja struje jednopolnog zemljospoja, što omogućava samogašenje prolaznog zemljospoja.
Konkretno u ovom primjeru izmjerena struja kvara iznosi 35,85A, manja je od 40A, tako da je omogućeno samogašenje zemljospoja. Međutim, rezultati dobijeni impedantnim algoritmom su utoliko bolji ukoliko su struje kvarova veće. Impedantni algoritam je primjenljiv u svim mrežama, osim za jednopolne kvarove u mrežama u kojima su zvjezdišta transformatora izolovana. Na slici 5.4 grafički su predstavljeni dobijeni rezultati proračuna lokacije kvara impedantnim algoritmom, tokom jednopolnog kvara, u slučaju izolovane mreže.



Slika 5.4 – Grafički prikaz rezultata impedantnog algoritma za mrežu sa izolovanim zvjezdištem

6. ZAKLJUČAK

U radu je obrađena impedantna metoda kao jedna od najzastupljenijih za lokaciju kvarova u DM, zasnovana na principima koji se primjenjuju kod distantne zaštite. Glavna prednost ove metode je mogućnost izostavljanja komunikacije, jer sa zadovoljavajućom tačnošću koristi mjerenu impedansu samo sa jednog kraja, kao i mogućnost jednostavne implementacije u digitalne snimače i zaštitne releje. Ekonomska opravdanost primjene neke od tehnika za lociranje kvara i tehnička opremljenost mreže mogu da se uzmu kao početna tačka pri strateškim odlukama tokom izbora metoda za lokaciju kvara u DM.

7. LITERATURA

[1] M.M. Saha, J. Izykowski, E. Rosolowski: *Fault Location on Power Networks*, Springer-Verlag, London, 2010.

[2] L. Jian: *Fault Location and Service Restoration for Electrical Distribution Systems*, Singapore: John Wiley & Sons, Inc., 2016.

[3] D. Bekut: *Relejna zaštita*, Fakultet Tehničkih Nauka, Novi Sad, 2009.

[4] S. Hanninen, M. Lehtonen: *Earth fault distance computation with fundamental frequency signals based on measurements in substation supply bay*, Reasearch notes, Espoo, 2002.

[5] S. Hanninen, *Single phase earth faults in high impedance grounded networks, Characteristics, indication and location*, Doctoral dissertation, Helsink University of Technology, Finland, 2006.

Kratka biografija:

Slavka Trakilović rođena je u Tuzli 1991. godine. Osnovne studije završila je na Fakultetu tehničkih nauka 2016. godine iz oblasti Elektrotehnike i računarstva – Elektroenergetski sistemi. Master rad na istom fakultetu smjer Elektroenergetika – Elektroenergetski sistemi odbranila je 2019. godine.

Зборник радова Факултет техничких наука, Нови Сад

UDK: 621.31
DOI: https://doi.org/10.24867/07BE19Adamovic

КОМУНИКАЦИОНИ ПРОТОКОЛИ У АУТОМОБИЛСКОЈ ИНДУСТРИЈИ И АУТОМАТИЗАЦИЈИ

COMMUNICATION PROTOCOLS IN AUTOMOTIVE INDUSTRY AND AUTOMATION

Милош Адамовић, *Факултет теничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *Предмет овог рада су најчешће коришћени комуникациони протоколи у аутомобилској индустрији и аутоматизацији. Прво ће бити дат кратак историјат развоја аутомобилске индустрије као и електричних и електронских система који се користе у управљању. Од посебног значаја за рад јесу CAN, LIN и CANopen протоколи који су детаљно описано уз примере њихове примене у возилима и индустрији. На самом крају извршиће се поређење поменутих, дати одређене предности и мане, као и могућност даљег развоја и унапређења.*

Кључне речи: *Аутомобилска индустија, CAN у аутоматизацији, Комуникациони протоколи*

Abstract – *The subject of this study is an analysis of the communication protocols in automotive industry and automation. First of all brief history of the development of the automotive industry as well as the electrical and electronic systems used in control will be given. The major importance for study are communication protocols CAN, LIN, CANopen which are detaile described with examples for every each. On the end comparison between mention communication protocols with advantages and disadvantages and as well as the possibility of further development and improvement will be made.*

Keywords: *Automotive industry, CAN in Automation, Communication protocols*

1. УВОД

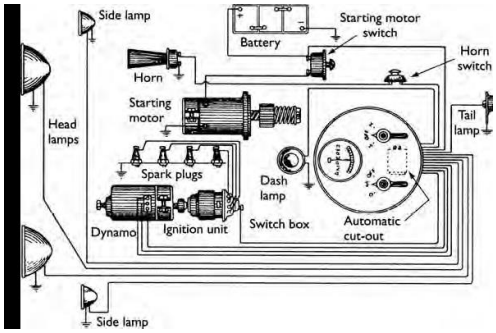
Услед изузетно брзог развоја аутомобилске електронике, комплексности система који се у данашње време уграђују у савремене аутомобиле, као и због великог броја различитих ECU јединица које се користе за управљање тим системима постоје потребе за што бржим, ефикаснијим, сигурнијим преносом података између истих.

Потребан је рад у реалном времену и нема простора за грешке приликом рада. Предмет овог рада представља опис и примену различитих комуникационих протокола у аутомобилској индустирији и аутоматизацији. Посебан осврт ће бити на употребу CAN протокола у аутомобилској индустрији као и његова широка примена у аутоматизацији.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Дарко Марчетић, ред. проф.

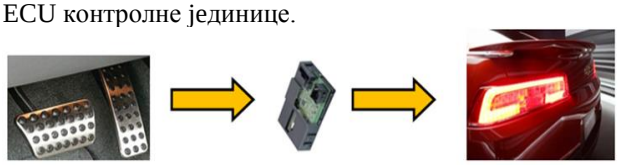
2. АУТОМОБИЛСКА ИНДУСТРИЈА

Развој аутомобила какве данас познајемо је почео 1885. године. Познати конструктори су се утркивали ко ће направити бољи патенат и увести неку новину што је и данас случај.
Руку под руку са развојем СУС мотора је ишао и развој електричних, а касније и електронских система. Прва електрична светла у возилима су уграђена 1908. године, а постају стандардни део опреме 1920. године.



Слика 1.1. *Комплетна електрична шема типичног аутомобила из '30. година XX века*

Најранији елекртични системи коришћени у возилима су се појавили почетком XX века и након Другог светског рата долази до наглог развоја и примене електронике у аутомобилској индустрији да би се дошло до електричних система који се налазе на хибридним возилима и електричним аутомобилима. Системима на данашњим возилима управљају моћне ECU контролне јединице.

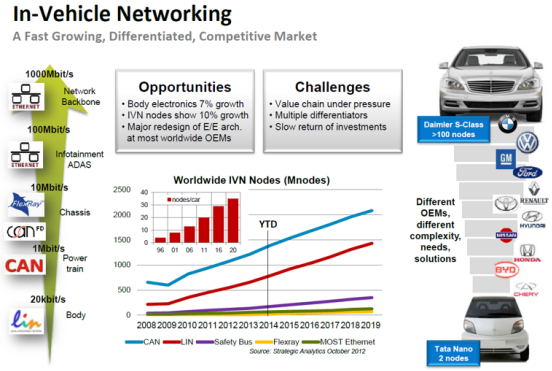


Слика 1.2. *Пример једноставног управљања које врши ECU контролна јединица.*

3. КОМУНИКАЦИОНИ ПРОТОКОЛИ

Као што је наглашено у уводу, комуникациони протоколи који се користе у аутомобилској индустрији треба да задовоље специфичне карактеристике преноса података..
У зависности од класе у којој се налазе, модерна возила могу да имају преко стотину ECU контролих јединица. Данас најзаступљенији комуникациони протоколи који се користе у савременим возилима су

CAN, LIN, MOST као и FlexRay, Ethernet који се користе у моћнијим системима где се захтева велика брзина и проток информација.

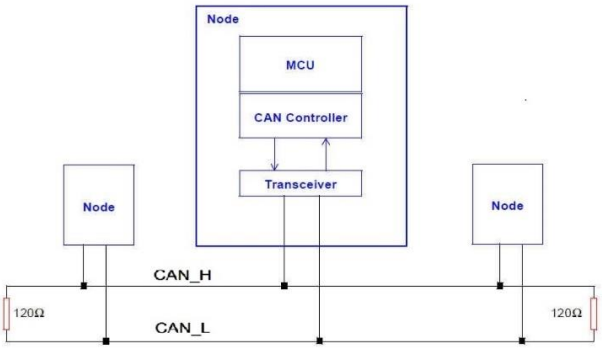


Слика 3.2 *Примена комуникационих протокола у савременим возилима*

CAN протокол са својим разним побољшаним варијатама као нпр. CAN FD је тренутно најзаступљенији у аутомобилској индустири. Свакако један од најбитнијих је LIN протокол користи се на оним местима у контролном систему возила где брзина преноса није критична. MOST протокол је прилагођен за мултимедију возила. FlexRay је протокол који има изразито боље карактеристике од CAN протокола и у будућности ће га сигурно заменити. Ethernet протокол је такође заступљен управо због све више компликованијих за управљање и све више заступљенијих система који треба да обезбеде аутономну вожњу као што је ADAS систем.

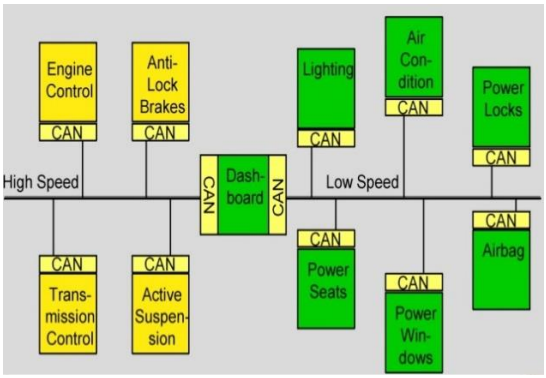
3.1. CAN протокол

Robert Bosch GmbH је 1982. године увео „CAN bus“ и тако заменио дотадашњу комуникацију увођењем јединствене магистрале. Први аутомобил који је поседовао CAN магистралу и који је ушао у масовну производњу био је Mercedes-Benz S-Class из 1991. године. На обједињеној магистрали је комуницирало 5 ECU контролних модула. ISO (International Organization for Standardization) је 1993. године је прихватио CAN протокол кака стандард ISO 11898.
CAN је асинхрона „multi-master“ и „bidirectional half duplex“ серијска магистрала која је реализована помоћу пара уплетених жица и раздвојена отпорником различитих отпорности зависно од конкретног стандарда. Магистралу чине два или више чворова прикључених на линеарну мрежу.



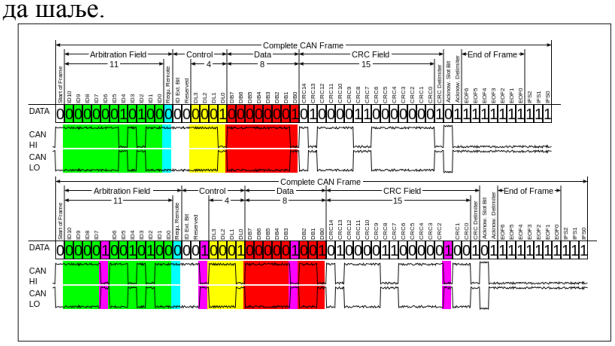
Слика 3.1.1. *Шематски приказ CAN чворова на магистрали (Node-ECU контролна јединице)*

Возило може да поседује две или више CAN мрежа које раде на различитим брзинама. У зависности од потребе за брзином постоје Low-speed CAN и Higher-speed CAN.



Слика 3.1.2. *Пример умрежавања електронских компоненти у возилу применом CAN магистрале*

Модули у случају CAN протокола комуницирају тако што користе две одвојене жице за комуникацију, те жице се називају CAN H (од „CAN High“) i CAN L (од „CAN Low“), види слику 3.1.1. Пошто се комуникација ослања на напонску разлику између две магистралне линије, CAN сабирница није осетљив на индуктивне скокове,друга електрична поља или друге сметње. То чини CAN магистралу поузданим избором за мрежну комуникацију у системима који су потребни на возилима, као и у осталим индустријским системима.
Како је већ поменуто CAN протокол је асинхрона „multi-master“ мрежа заснована на порукама где сваки чвор има могућност да шаље и прима поруке. Да не би дошло до колизије на магистрали постоје правила по којима предност имају оне поруке које су вишег приоритета и због тога је дефинисано да на једној мрежи сваки чвор мора да има скуп порука које може да шаље.

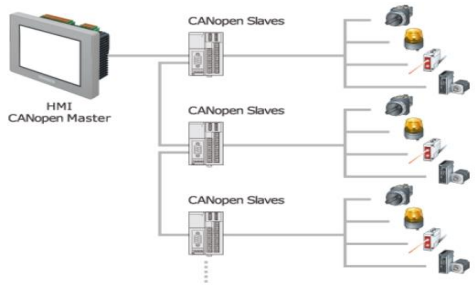


Слика 3.1.2.3. *Порука у основном формату са приказаним електричним нивоима*

3.2. CANopen протокол

Индустријске магистрале у индустријским мрежама се користе за повезивање „Field“ уређаја. Такве мреже су: ProfiBus, PROFInet, Modbus, Ethernet и многе друге, али оне нису предмет овог рада већ употреба CAN протокола у комбинацији са поменутим. Као што је написано поред „Fieldbus“ постоје и мреже засноване на CAN протоколу које чине основни CAN и виши CAN протоколи у које спада и CANopen.

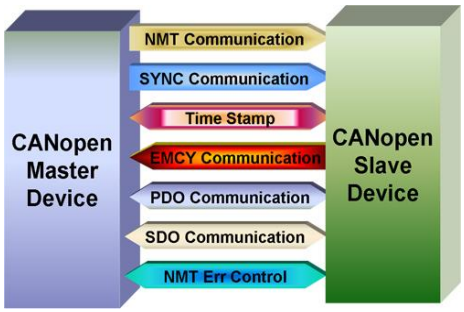
CANOpen магистрала је у већини случајева имплементирана у оквиру неког великог система и одговарајућим „gateway“ уређајима повезана са остатком тог великог система.



Слика 3.1.4.1. *Приказ примене CANopen протокола у индустријском управљању*

CANopen омогућава 127 чворова на магистрали , док је избор могућих брзина преноса података далеко већи. На мрежи мора постојати бар два чвора од којих у сваком тренутку један мора бити „Master“, а други „ Slave“.

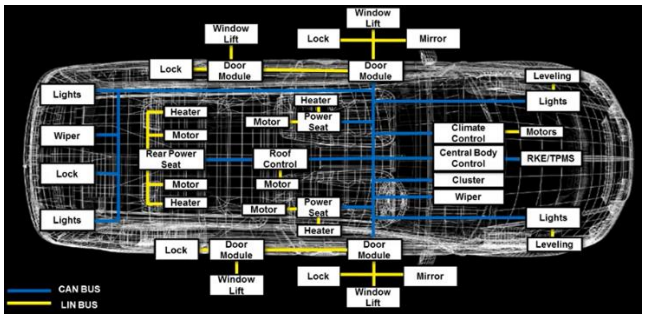
CANopen протокол врши сегментацију великих порука у 8-битне пакете и шаље их пакет по пакет у оквиру поруке дужине 111 битова. То омогућава да порука вишег приоритета увек прекине пренос велике поруке.CANopen уређаји у подржани са сервисом за администрирање на мрежи помоћу CANopen network management (NMT) slave state machine и она дефинише начин комуникације између урађаја на мрежи.



Слика 3.1.4.2. *Приказ CANopen комуникације*

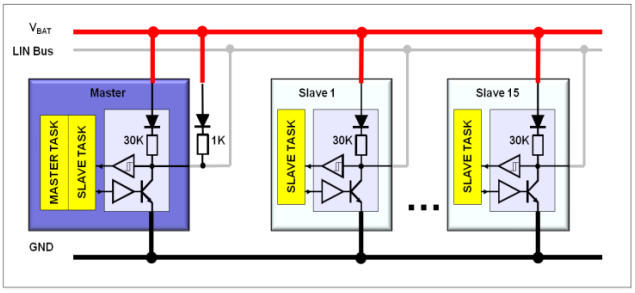
3.2. LIN протокол

LIN мрежа се у аутомобилској индустрији користи као подмрежа CAN мреже, а може и независно да буде директно повезана на главни рачунар возила. Такође постоје одређени „Gateway“ уређаји који могу један протокол да прилагоде другом.



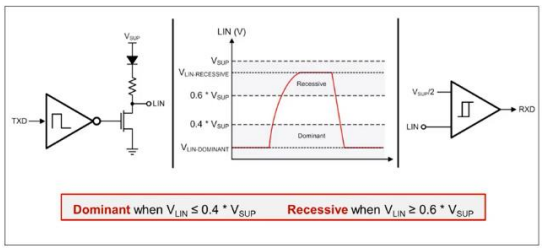
Слика 3.2.2. *Приказ LIN протокола у савременим аутомобилима*

LIN магистрала користи једну жицу за комуникацију и у оквиру LIN протокола постоји само један „Master“ и до 15 „ Slave“ уређаја. Пренос података је синхрони и максималном брзином од 20Kbit/s. Физички ниво LIN протокола је заснован на стандарду ISO 9141 са неким модификацијама за возила. У данашњим топологијама умрежавања користе се микроконтролери који имају или UART могућност или намјенски LIN хардвер.



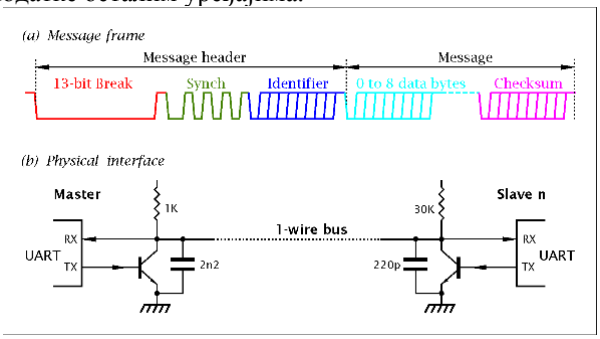
Слика 3.2.1.1. *Шематски приказ физичког нивоа LIN магистрале*

„Master“ и „ Slave“ уређаји у свакој њиховој поруци помоћу примопредајника који претвара бит логике из микроконтролера у више напонске нивое и на тај начин формира поруку која је препознатљива свима на магистрали.



Слика 3.2.1.2. *Вредности напона LIN магистрале приликом примања и слања битова*

Све поруке на магистрали контролише „Master“ тако што шаље и захтева поруке које у себи садрже заглавље поруке (Message Header) и део поруке са одговором (Message Response).). Одговор креирају „ Slave“ уређаји који су прозвани, али може и сам „Master“ уколико је то потребно да пошаље одређене податке осталим уређајима.



Слика 3.2.2.1. *Приказ физичке архитектуре и изгледа LIN поруке која се шаље на магистралу*

3.3. Кратак преглед и основне особине комуникационих протокола који су тема овог рада

Како је већ наглашено, различити комуникациони протоколи се користе у различитим системима и на различитим деловима тог система.
Одлуку где ће се користити који протокол углавном доносе произвођачи у складу са потребама система и финансијским могућностима.

Тип магистрале	CAN	LIN	Flexray	MOST	Ethernet
Брзина преноса	1Mbit/s	20Kbit/s	10 Mbit/s	24 Mbit/s	10 Gbit/s
Повезивање	2 жице	1 жице	2 i 4 жице, оптички кабел	оптички кабел	2 жице
Архитектура	Multi-Master, до 64 чвора	Single-Master ,до 16 чвора	Multi-Master ,до 64 чвора	Multi-Master ,до 64 чвора	Multi-Master, неограничен број
Начин комуникације	Асинхрони	Синхрони	Асинхрони и Синхрони	Асинхрони и Синхрони	Синхрони
Надоградња	могућа	могућа	могућа	могућа	могућа
Употреба	Контрола мотора и конфора	Конфор	Контрола мотора и сигурности	CD и DVD, GPS навигација	360° камера и мултимедија
Cena	srednja	niska	visoka	srednja	visoka

Табела 3.3.1. *Приказ поређења комуникационих поротокола у аутомобилској индустрији*

4. МОДЕРНИ СИСТЕМИ НА ВОЗИЛИМА И СПЕЦИЈАЛНИ ЗАХТЕВИ

Опрема у возилу се с временом развила тако да укључује разне аутомобилске системе нове генерације који захтевају специјалну повезаност, попут напредног система аутономне вожње и помоћи возачу - advanced driver assistance systems (ADAS) и многи други.
Ови системи брзо постају стандардна опрема на многим возилима данашњице.
Поред тога, услед страховитог пораста истраживања и развоја аутономних возила, електронска опрема има задатак да испоручује сигнале велике брзине, велике пропусности и велике снаге за управљање тим системима.
Такође захтева се да комуникациони протоколи у овим системима испуне све задатке који се стављају пред њих велика брзина, велика пропусна моћ локалне мреже као и у новије време потребе за комуникацион са спољним светом, GPS-ом, осталим возилима у околини и саобраћајним системима

5. ЗАКЉУЧАК

Моторна возила последњих година су кренула вртоглаво да се развијају. Сам управљачки систем мора да одговори на све веће захтеве. Комплексност система диктира цену возила и произвођачи истих се боре да смање цену производње, како би били конкурентни на тржишту.
У системима данашњих возила да би била задовољена функционалност и безбедност потребни су комуникациони протоколи који ће на ефикасан и брз начин повезати цео систем. Рад у реалном времену је захтев број један који треба испунити. Комуникациони протоколи који контролишу пренос података морају да имају велику пропусну моћ и брзину преноса података.

6. ЛИТЕРАТУРА

[1] *Tom Denton, Automobile Electrical and Electronic Systems, Elsevier Butterworth-Heinemann, 2004*

[2] *Dr. Wilrid Dubitzky, Turgut Karacay, "CAN – From its early days", CAN Newsletter, 1/2013*

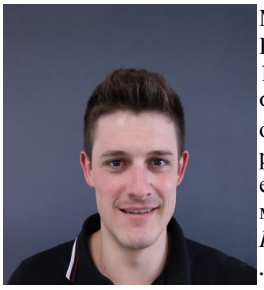
[3] *J. Lee, K. Choi, and J. Lee, "Collecting Big Data from Automotive ECUs beyond the CAN Bandwidth for Fault Visualization", Hindawi Mobile Information Systems Volume 2017, Article ID 4395070, 13 pages, 27 February 2017*

[4] *Марчетић,,,Микропроцесорско управљање енергетским управљачима" ФТН Издаваштво, Нови Сад, Србија*

[5] *Веб сајт:* https://www.automatika.rs/baza-znanja/obrada-signala/canopen-visi-can-protokol.html

[6] *Веб сајт:* https://www.can-cia.org/can-knowledge/can/systemdesign-can-physicallayer/

Kratka biografija:



Милош Адамовић рођен је у Београду, Република Србија 17.4.1993. Године. Мастер рад на Факултету техничких наука из области Електротехнике и рачунарства - Енергетска електроника и електричне машине одбранио је 2019. године.
Контакт:adamovicmilos993@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/07BE20Vidric

ANALIZA RADA RELEJNE ZAŠTITE U SEVERNOAMERIČKIM DISTRIBUTIVNIM MREŽAMA

ANALYSIS OF RELAY PROTECTION OPERATION IN NORTH AMERICAN DISTRIBUTION NETWORKS

Dragana Vidrić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu opisane su osnovne karakteristike severno američkih distributivnih mreža i nove tendencije razvoja i upetljavanja u svrhu sigurnosti. Navedeni su opšti zahtevi koji se postavljaju pred relejnu zaštitu i date su osnovne informacije o prekostrujnim i distanitnim relejima. Predstavljene su funkcije relejne zaštite. Na primeru jednopolne šeme severnoameričke distributivne mreže, prikazano je rešenje pri promeni uklopnog stanja mreže.*

Ključne reči: *srednjenaponske nebalansirane distributivne mreže, principi rada relejne zaštite, adaptivna relejna zaštita, analiza osetljivosti relejne zaštite*

Abstract – *This paper describes the basic characteristics of North American distribution networks and the new tendencies of development and interference for safety purposes. General requirements for relay protection and basic information on overcurrent and distance relays are given. The relay protection functions are presented. In the example of a single-pole North American distribution network, a solution is shown when changing the switching state of the network.*

Keywords: *distribution unbalanced networks, functionality od relay protection, adaptive relay protection, relay protection sensitivity analysis*

1. UVOD

Osnovni cilj primene relejne zaštite je najbrže moguće isključenje elementa, odnosno dela elektroenergetskog sistema prilikom nekog abnormalnog stanja, inicijalizacijom odgovarajućih upravljačkih akcija, uz očuvanje funkcionalnosti i održanje stabilnosti ostalog dela sistema.
Releji moraju izolovati samo deo mreže pod kvarom, omogućavajući nastavak rada nad delom mreže bez kvara.
Time se omogućava nesmetan rad mreže u kojoj su instalisani releji. Srednjenaponske distributivne mreže odlikuju se radijalnim ili slaboupetljanim pogonom, gde se različite vrste relejnih zaštita primenjuju [2].
U glavi 2 dat je opis osnovnih karakteristika distributivnih mreža i nove tendencije razvoja i upetljavanja u svrhu sigurnosti.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je dr Duško Bekut, redovni profesor.

Takođe, dat je uporedni prikaz dva osnovna tipa mreža u svetu: evropskih i severnoameričkih. U glavi 3 i 4 se opisuju najčešće vrste relejne zaštite koje se koriste u radijalnim i slaboupetljanim distributivnim mrežama – prekostrujna i distantna, kao i detaljan opis konkretnih tipova i funkcionalnost releja koji se koriste u severnoameričkom distributivnom području. U petoj i šetoj glavi je izložen praktičan deo rad, data je jednopolna šema distributivne transformatorske stanice, kao i predložena softverska rešenja koje daje funkcija Adaptivna relejna zaštita, pri promeni usklopnog stanja mreže. U sedmoj i osmoj glavi je naveden zaključak rada i korišćena literatura, respektivno.

2. OSNOVE DISTRIBUTIVNIH MREŽA

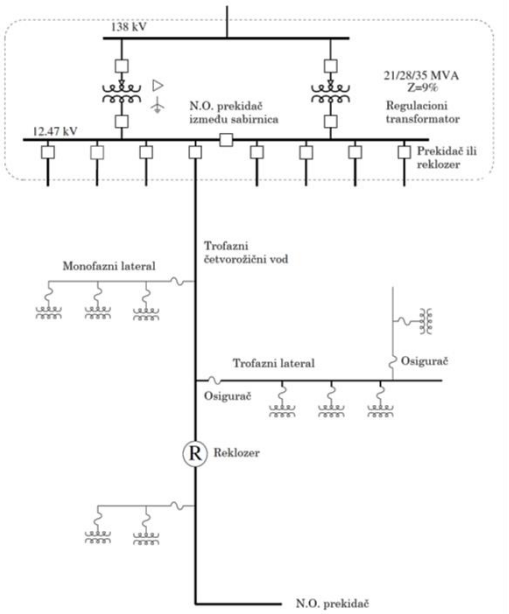
Elektrifikacija koja je započela početkom 20–tog veka suštinski je doprinela produktivnosti i industrijalizaciji sveta [2]. U početku nije bilo mnogo potrošača, ali je njihov broj ipak bio preveliki da bi se distributivne mreže mogle upetljavati i da bi se tako upetljavanje u svrhu sigurnosti finansijski isplatilo. Vremenom se broj potrošača povećava i sama struktura potrošača se menja – postoji sve više uređaja u običnim domaćinstvima, motori u industrijskim zonama imaju sve veće snage. To je dovelo do usložnjavanja distributivnih mreža i konačnog opredjeljenja za radijalnost kao osnovu funkcionalnosti ovakvih mreža [3]. Ipak, neka potrošačka područja su zahtevala veći stepen sigurnosti i pouzdanosti pa se tamo pristupilo maloj modifikaciji mreža – uveden je koncept slabo upetljanosti u takvim područjima

2.1 Upetljane distributivne mreže

Kao što je spomenuto, jedan od najefikasnijih načina za rešavanje problema ispada potrošača je upetljavanje mreža. Područja koja su od velikog ekonomskog ili društvenog značaja kao što su bolnice, aerodromi, vojni objekti ili jednostavno gusta potrošačka područja zahtevaju konstantno napajanje, da ne bi došlo do većih socijalnih, društvenih ili ekonomskih posledica. U svetu ne postoje jedinstvena rešenja za navedene probleme, pa tako i ne postoji obrazac konfiguracije distributivnih mreža. Generalno, kao posebni tipovi distributivnih mreža izdvojili su se evropski i severnoamerički tip [4].
Elektroenergetski sistemi u Evropi se u praksi razlikuju u zavisnosti od država i zakonskih propisa. Opšte smernice prilikom konfiguracije sistema su da se prenosne mreže upetljavaju radi zadovoljenja sigurnosti tipa „n-1“, dok se distributivne mreže izvode kao radijalne, ukoliko struktura potrošačkog područja ne zahteva suprotno.

Ukoliko postoji potreba za upetljavanjem, obično se u evropskim mrežama izvodi neuporedivo manji broj petlji. Zbog toga postoji i razlika u terminologiji pa se ovakve mreže nazivaju slaboupetljane distributivne mreže (eng. *Weakly Meshed Network*) [4].

Struktura čitavog elektroenergetskog sistema Severnoj Americi se razlikuje u odnosu na evropske mreže, ali te razlike posebno dolaze do izražaja u distributivnim mrežama. Distributivne stanice se obično izvode kao sistem dva transformatora u bloku. Ovi transformatori su dimenzionisani da mogu da preuzmu opterećenje druge stanice ukoliko dođe do njenog ispada. Ove stanice obično sa napona 230 kV ili 35 kV prebacuju na napon mrežnog transformatora, koji je u opsegu 4 kV – 35 kV. Mrežni transformatori spuštaju napon na potrošački, priključni napon (120 V ili 240 V). Na slici 2.1 dat je jedan primer ove konfiguracije. Izvod prikazan na slici je samo jedan od mnoštva koji se napaja sa date stanice. On je izveden kao nadzemni vod ili podzemni kabl velike prenosne moći, u zavisnosti od stepena urbanizma potrošačkog područja. Sa glavne grane izvoda granaju se laterali koji mogu biti različitih faznosti i opterećenja i zaštićeni su topljivim osiguračima ili prekidačima. Ovakvi laterali su srednjenaponski i na njima se nalazi nekoliko mrežnih transformatora. Ovi transformatori se razlikuju od evropskih jer su uglavnom monofazni; u slučaju da su trofazni izvode se kao tri odvojena monofazna transformatora.



Slika 2.1 – *Primer distributivne stanice/mreže severnoameričkog tipa*

3. ZAŠTITA U RADIJALNIM DISTRIBUTIVNIM MREŽAMA

Krajem devetnaestog i početkom dvadesetog veka za zaštitu elektroenergetskog sistema počinju da se koriste releji, gde se pod pojmom relej podrazumeva uređaj koji je napravljen tako da, kada se na njegove ulaze dovodi električna, mehanička ili neka druga veličina odgovarajućeg intenziteta, deluje na neki unapred određen način. Pri tome, osigurači i slični uređaji ne spadaju u releje, jer je njihova funkcionalnost nepovratno izgubljena nakon jednog delovanja [2]. Relej je uređaj koji služi za detekciju abnormalnih pogonskih stanja elemenata i delova sistema, i na osnovu toga vrši inicijalizaciju odgovarajućih upravljačkih akcija za obezbeđenje normalnog pogona. Pri ispunjenju ovog cilja potrebno je ispuniti nekoliko opštih zahteva: selektivnost, brzina reagovanja, osetljivost, pouzdanost, cena itd.

Relejna zaštita može biti klasifikovana u skladu sa načinom na koji obavlja svoju funkciju. Postoje: prekostrujni, distantni, diferencijalni, podnaponski, drugi releji. U nastavku rada je analizirana prekostrujna i distantna zaštita.

3.1 Prekostrujna zaštita

Prekostrujna zaštita generalno predstavlja najčešći oblik zaštite koja se koristi za eliminisanje kvarova u mreži, detektovanjem velikih vrednosti struje kvara. Prekostrujni releji su najrasprostranjenija vrsta zaštite zbog relativno niske cene i širokog opsega struje kvara koje mogu detektovati. Na osnovu radne karakteristike releja, postoji podela na tri glavne grupe: trenutni prekostrujni releji, releji sa strujno nezavisnom karakteristikom i releji sa strujno zavisnom karakteristikom.

3.2 Distantna zaštita

Rad distantnih releja nije regulisan striktno strujom ili snagom, već zavisi od odnosa primenjenog napona i struje. Releji stoga efektivno mere impedansu u zoni koju štite. U uslovima kvara, napon je degradiran, i tada se struja u velikom intenzitetu povećava. Ovo se detektuje distantnim relejom kada impedansa opadne tokom kvara. Iniciran je prekid napajanja, koji se obavlja slanjem signala strujnom prekidaču da prekine strujno kolo. Ovaj tip releja (*eng. distance relays*) koristi se u nadzemnim i kablovskim mrežama. Prednost im se ogleda u brzini, kada druge zaštite nisu u mogućnosti da ispoštuju zahtevane vremenske uslove. Impedansom može da se odredi dužina nadzemnog voda ili kabla, te se distantna zaštita naziva i zaštita distantom impedansom (*eng. Distance impedance*).

Kada se posmatra karakteristika distantnog releja, razlikuju se tri tipa: impedantna, ugaono-admitantna i kvadrilateralna.

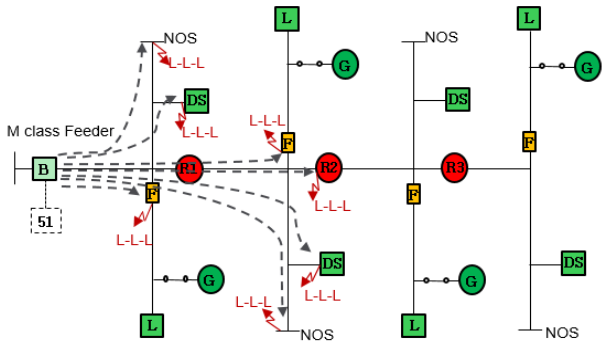
4. REALIZACIJA RELEJNE ZAŠTITE U SEVERNOAMERIČKIM DISTRIBUTIVNIM MREŽAMA

U ovom radu, realizacija relejne zaštite se ogleda u korišćenju dve funkcije – Adaptivna relejna zaštita (u daljem tekstu ARZ) i Analiza osetljivosti relejne zaštite (u daljem tekstu AORZ), koje su deo slobodno dostupne trajal verzije softvera za proračune kratkih spojeva i osetljivosti zaštite. ARZ se koristi za trenutni odabir najbolje grupe (podešenja) zaštite u okviru zaštitne opreme (engl. *Protection equipment*), dok se AORZ koristi za planiranje. Zaštitna oprema se nalazi na reklozerima i prekidačima koji se nalaze na samom početku izvoda (*engl. feeder head breaker*), s tim da se tipovi releja koji se na njima nalaze razlikuju. Prekidači (na samom početku izvoda) mogu da imaju oba tipa zaštite (i prekostrujnu i distantnu), dok reklozeri koji se nalaze u dublje u mreži imaju samo prekostrujne releje na sebi. Svaka zaštitna oprema može imati jednu ili više grupa sa različitim podešenjima. U nastavku rada biće izloženi primeri tri najčešće korišćena releja u severnoameričkim distributivnim mrežama, kao i opisana njhova funkcionalnost rada.

4.1 „51“ – vremenski fazni prekostrujni relej

Štiti objekte od tropolnih kratkih spojeva, s tim da pokriva čitavu primarnu i sekundarnu zonu, preciznije seže do normalno otvorenog prekidača, distributivne stanice, osigurača i drugog po redu reklozera – slika 4.1.

Algoritam pomenutih funkcija (ARZ i AORZ) se bazira na tome da se na tim mestima simulira tropolni kratak spoj i od svih vrednosti struja bira se najmanja, koja se dalje upoređuje sa podešenjem releja.



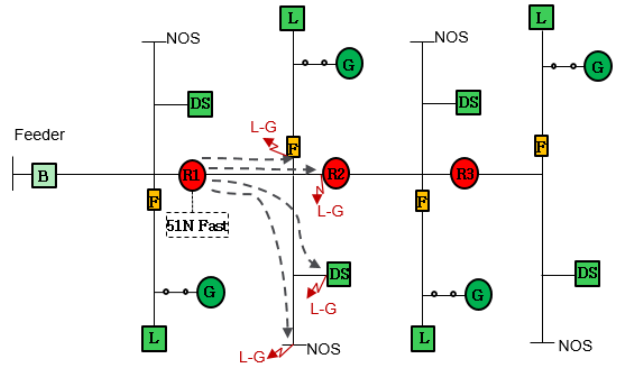
Slika 4.1 – *Princip rada 51 releja*

Na slici su korišćene sledeće oznake:
B – prekidač na početku izvoda,
R1, R2, R3 – reklozeri,
L – potrošač,
NOS – normalno otvoren prekidač,
DS – distributivna stanica
L-L-L – tropolni kratak spoj,
L-G – zemljospoj,
G – distributivni generator ili sinhrona mašina,
F – osigurač.

4.2 „51N FAST RECLOSER“ zaštita

Za razliku od prethodnog releja, ova zaštita se postavlja na reklozerima, otuda i ime. Štiti objekte od jednopolnih kratkih spojeva, a pritom pokriva 100% primarne zone, tj seže do normalno otvorenog prekidača, distributivne stanice, prvog susednog reklozera i osigurača (slika 4.2).

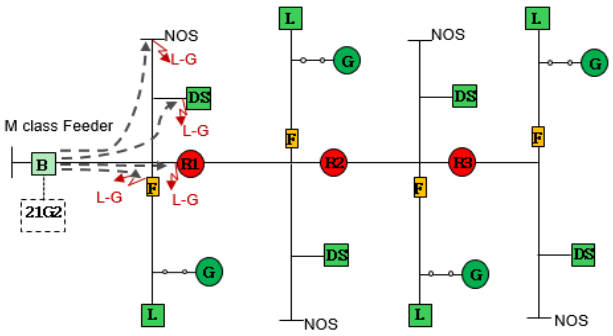
Algoritam funkcija se bazira na tome da se na tim mestima simuliraju jednopolni kratki spojevi i bira se najmanja vrednost struje kratkog spoja, koja se upoređuje sa podešenjem releja. Vreme delovanja je 0.1s.



Slika 4.2 – *Princip rada „51N FAST Recloser“ zaštite*

4.3 „21G2“ – zemljospojni distantni relej (zona štićenja 2)

Štiti objekte od jednopolnih kratkih spojeva. Pokriva 100% primarne zone, tj seže do normalno otvorenog prekidača, distributivne stanice, reklozera i prvog osigurača (Slika 4.3). Na tim mestima se simuliraju jednopolni kratki spojevi i bira se najmanja vrednost struje kratkog spoja, koja se upoređuje sa podešenjem releja. Ima kvadrilateralnu karakteristiku.



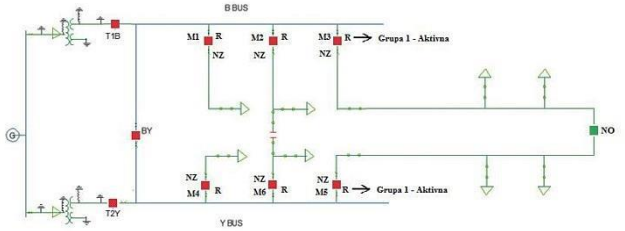
Slika 4.3 – *Princip rada 21G2 releja*

5. ADAPTIVNA RELEJNA ZAŠTITA

Adaptivna relejna zaštita predstavlja izuzetno važan koncept koji je nastao kao rezultat uvođenja mikroprocesorskih zaštita, inteligentnih elektronskih uređaja i naprednih komunikacionih sistema. U samoj sustini adaptivne zaštite, nalazi se centralizovani algoritam, koji funkcioniše na bazi poređenja aktuelnih podešenja zaštite i trenutnog stanja zaštita u mreži. Na ovaj način, detektuju se zaštitni uređaji sa neadekvatnim podešenjima i fokus se stavlja na traženje optimalne zaštite, ukoliko je trenutna zaštita mreže neadekvatna. Adaptivna relejna zaštita, kao deo slobodno dostupne trajal verzije softvera za proračune kratkih spojeva i osetljivosti zaštite, najčešće se pokreće za izvode ili celu transformatorsku stanicu. Nakon uspešnog izvršenja funkcije, dostupan je izveštaj o svim neophodnih informacijama.

5.1 Praktična realizacija Adaptivne relejne zaštite

Distributivna transformatorska stanica se nalazi na području Severne Amerike. Na jednopolnoj šemi prikazanoj na slici 5.1 mogu se primetiti dva tronamotajna transformatora koja rade u paraleli. Sistemi glavnih sabirnica na kojima se nalaze odgovarajući izvodi sa prekidačkom opremom, relejnom zaštitom i trenutnom aktivnom grupom (podešenjem) releja.



Slika 5.1 – *Jednopolna šema distributivne transformatorske stanice u inicijalnom stanju*

Gde su:
NZ – normalno zatvoreni prekidač,
NO – normalno otvoreni prekidač,
M1, M2… – imena izvoda.

Izveštaj ARZ-a za zaštitnu opremu na M3 izvodu izgleda kao u tabeli 5.1.

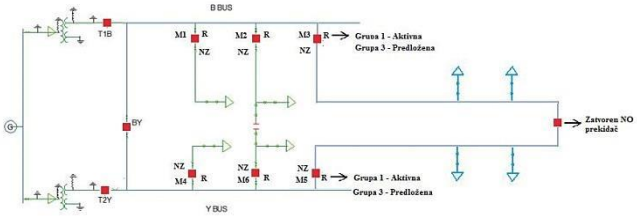
Tabela 5.1. *Izveštaj ARZ-a pri inicijalnom stanju mreže sa slike 5.1*

Tip releja	Zona	Osetljivost			Vreme delovanja	Opterećenje		
		Podešenje releja	Proračunata vrednost	Koeficijent efikasnosti		Podešenje releja	Proračunata vrednost	Koeficijent efikasnosti
ZAŠTITNA OPREMA - M3 IZVOD								
GRUPA 1 - AKTIVNA								
51	0	-	-	-	-	-	-	-
21P	2	80.00 Ω	26.21 Ω	●	-	80.00 Ω	353.03 Ω	●
21P	3	42.20 Ω	26.21 Ω	●	2.8s	42.20 Ω	353.03 Ω	●
51N	0	-	-	-	-	-	-	-
21G	2	80.00 Ω	27.20 Ω	●	-	-	-	-
21G	3	42.23 Ω	27.29 Ω	●	2.8s	-	-	-
GRUPA 2								
GRUPA 3								
GRUPA 4								

Svaki relej može da se klasifikuje na osnovu sledećih kriterijuma:

1) Osetljivost – proverava se da li je razmatrani relej izvan određenih limita (specificiranih unapred zadatih vrednosti), za ranije određenu vrednost tipa i lokacije kvara.
2) Vreme delovanja – proverava se da li je vreme odsecanja kvara u specificiranim limitima. Vreme isključenja kvara mora biti takvo da obezbedi čišćenje kvara u što bržem vremenskom periodu.
3) Opterećenje – proverava se da li je maksimalno opterećenje elementa u specificiranim limitima. Opterećenje štićenog elementa mora biti u okviru zadatih ograničenja kako bi se izbeglo nepotrebno delovanje (okidanje) releja u slučaju velikih opterećenja u mreži.

Iz nepoznatih razloga dolazi do promene uklopnog stanja mreže, tj do zatvaranja NO prekidača prikazanog na slici 5.1. Zadatak ARZ-a je da za novu konfiguraciju mreže predloži koja grupa (tačnije podešenje releja) u okviru zaštitne opreme će biti najadekvatnija.



Slika 5.1 – *Jednopolna šema distributivne TS nakon promene uklopnog stanja mreže*

Nakon uspešnog izvršenja ARZ funkcije, novi izveštaj sa predloženom grupom i informacijama kako bi ta grupa štitila izvod, je dostupan i izgleda kao u tabeli 5.2.

Tabela 5.2. *Izveštaj ARZ-a nakon promene uklopnog stanja mreže*

Tip releja	Zona	Osetljivost			Vreme delovanja	Opterećenje		
		Podešenje releja	Proračunata vrednost	Koeficijent efikasnosti		Podešenje releja	Proračunata vrednost	Koeficijent efikasnosti
ZAŠTITNA OPREMA - M3 IZVOD								
GRUPA 3 – PREDLOŽENA								
51	0	216.00 A	995.02 A	●	1.31s	-	-	-
21P	2	81.57 Ω	26.21 Ω	●	-	-	-	-
21P	3	108.77 Ω	26.21 Ω	●	2.80s	-	-	
51N	0	60.00 A	507.87 A	●	2.51s	-	-	-
21G	2	81.77 Ω	27.22 Ω	●	-	-	-	-
21G	3	109.00 Ω	27.31 Ω	●	2.80s	-	-	-
LE	0	-	-	-	-	45.97 Ω	352.92 Ω	●
GRUPA 1 – AKTIVNA								
GRUPA 2								
GRUPA 4								
ZAŠTITNA OPREMA – M5 IZVOD								
GRUPA 3 – PREDLOŽENA								
51	0	212.00 A	681.04 A	●	1.78s	-	-	-
21P	2	83.48 Ω	10.63 Ω	●	-	-	-	-
21P	3	111.30 Ω	38.20 Ω	●	2.80s	-	-	-
51N	0	100.00 A	368.75 A	●	1.08s	-	-	-
21G	2	84.05 Ω	11.59 Ω	●	-	-	-	-
21G	3	112.08 Ω	41.61 Ω	●	2.80s	-	-	-
LE	0	-	-	-	-	33.28 Ω	400.55 Ω	●
GRUPA 1 – AKTIVNA								
GRUPA 2								
GRUPA 4								

6. ZAKLJUČAK

U ovom radu obrađeni su načini realizacije relejne zaštite u severnoameričkim distributivnim mrežama. Detaljno su opisani tipovi releja, kao i njihova funkcionalnost rada. Kroz test primer, posredstvom jednopolne šeme distributivne transformatorske stanice, prikazano je najadekvatnije rešenje koje predlaže ARZ funkcija, pri promeni uklopnog stanja mreže.

U mrežama gde se, u zavisnosti od uklopnog stanja, struje kratkih spojeva, ali i nominalne struje, značajnije menjaju po intezitetu, može doći do neselektivnog delovanja ili do čak vrlo sporog delovanja zaštite. Iz tog razloga je neophodno precizno konfigurisati podešenje relejne zaštite s ciljem da se takvi problemi prevaziđu.

Ovaj rad se u tu svrhu može koristi kao polazna osnova za dalja istraživanja i proračune sa aspekta relejne zaštite, kao i integrisanje u druga softverska rešenja.

7. LITERATURA

[1] D. Bekut, *Relejna zaštita*, Fakultet Tehničkih Nauka Novi Sad, 2009.
[2] V. Strezoski, *Elektroenergetski sistemi*, Fakultet Tehničkih nauka, Novi Sad 2010.
[3] M. Nimrihter, *Distributivni sistemi*, Fakultet tehinčkih nauka, Novi Sad, 2001.
[4] T. A. Short, *Electric Power Distribution Handbook*, Second Edition, 2014.

Kratka biografija:

Dragana Vidrić je rođena u Zrenjaninu 1995. godine. Osnovne studije je završila na Fakultetu tehničkih nauka 2018. godine na departmanu Elektrotehnika i računarstvo – Elektroenergetski sistemi. Master rad je odbranila 2019. godine, na istom fakultetu na smeru Energetika, elektronika i telekomunikacije – Elektroenergetski sistemi.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/07BE21Dukic

ZAŠTITA TRANSFORMATORA

TRANSFORMER PROTECTION

Milica Dukić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu se razmatra zaštita transformatora. Cilj je da se opišu sve zaštite transformatora kao i primena mikroprocesorske zaštite. Razmatranja su ilustrovana na primeru dve transformatorske ćelije u 35 kV postrojenju. Razvijeno je rešenje u softverskom okruženju „SFT2841“ koje se koristi u slučaju primene zaštitnog uređaja SEPAM T87.*

Ključne reči: *zaštita transformatora, mikroprocesorska zaštita*

Abstract – *This paper deals with transformer protection. The goal is to describe all types of transformer protection and use of microprocessor protection. Considerations are illustrated on example with two transformer cells in 35kV switchgear. A software solution has been developed in „SFT2841“ software environment. This environment is used for a case of SEPAM T87 protection device.*

Keywords: *Transformer protection, Microprocessor protection*

1. UVOD

U radu se razmatra zaštita transformatora. Zaštita transformatora, kao i naravno bilo kog dela elektroenergetskog sistema, je veoma važan deo održavanja sistema. Ono na šta se u današnje vreme sve češća pojava je primena mikroprocesorske zaštite umesto starih elektromehaničkih zaštita. U poglavlju 2 kratko su opisani mogući kvarovi transforamtora i šta su mogući uzroci tih kvarova. U poglavlju 3 nabrojane su i opisane vrste zaštita koje se koriste za zaštitu transformatora. Navedena su i opasna pogonska stanja kao i zaštite koje se koriste u takvim situacijama. Poglavlje 4 posvećeno je mikroprocesorskoj zaštiti. Opisan je istorijski razvoj mikroprocesorske zaštite, princip rada i vrste ovih zaštita. Na kraju ovog poglavlja dato je poređenje mikroprocesorske zaštite i klasične zaštite. U poglavlju 5 dat je konkretan primer podešenja zaštite transformatora na transformatorskim ćelijama 35 kV postrojenja toplane „Cerak“. Kao zaštitni uređaj korišćen je relej SEPAM T87. Podešenja zaštite se realizuju u softverskom okruženju „SFT2841“. Poglavlje 6 tiče se ispitivanja zaštite. Objašnjeno je iz kojih razloga je neophodno ispitavanje funkcionalnosti zaštite pre puštanja u rad. U poglavlju 7 razmatra se sistem za upravljanje, zaštitu i nadzor – SCADA (Supervisory Control and Data Acquisition) sistem. Navedene su mnogobrojne prednosti korišćenja ovog sistema. U poglavlju 8 dat je zaključak.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je dr Duško Bekut, red. prof.

2. ZAŠTITA TRANSFORMATORA

Kvarovi na transformatorima prvenstveno nastaju kao posledica slabljenja i oštećenja izolacije. Oštećenja mogu biti izazvana naprezanjima električne prirode, mehaničke prirode ili kao posledica prevelikog zagrevanja. Kada govorimo o kvarovima transformarora, postoji zahtev za brzom eliminacijom kvarova jer postoji opasnost od pucanja kotla i paljenja ulja.
Obično se nailazi na sledeće kvarove:

1. Kratki spojevi između namota transformatora.
2. Kratki spojevi između navojaka iste faze.
3. Kratki spojevi sa zemljom.
4. Lokalna tinjanja izolacije zbog previsokih električnih naprezanja ili kao posledica smanjenja kvaliteta izolacije usled prevelikog zagrevanja [1].

3. VRSTE ZAŠTITA TRANSFORMATORA

Kao zaštite transformatora od kvarova koriste se sledeće zaštite:

- Diferencijalna.
- Gasna.
- Zemljospojna.
- Trenutna prekostrujna.
- Distantna.
- Osigurači.

Kod transformatora postoje i opasna pogonska stanja – spoljašnji kratki spojevi, preopterećenje i isticanje ulja iz kotla. Kao zaštite od opasnih pogonskih stanja koriste se prekostrujna zaštita, zaštita od preopterećenja i gasna zaštita.

3.1 DIFERENCIJALNA ZAŠTITA

Ova zaštita se koristi kada se transformatora štiti od kratkih spojeva između namota, navoja i kratkih spojeva sa zemljom. Kod primene ove zaštite nailazi se na problem razlike struja sa njegovih strana, po modulu i faznom stavu. Ovaj problem je dodatno složen ukoliko je fazni pomeraj sprege transformatora različit od 0 sati.

3.2 GASNA ZAŠTITA

Gasna zaštita sa Buholc relejom se može koristiti kod transformatora kod kojih postoji dilatacioni sud. Princip rada ove zaštite se zasniva na tome da pri pojavi kvarova unutar kotla dolazi do intenzovnog nastanka gasova i povećanja pritiska u kotlu. Konstrukcijom Buholc releja omogućena je i detekcija lokalnih tinjanja izolacije i kratkih spojeva.

3.3 ZAŠTITA OD KRATKIH SPOJEVA SA ZEMLJOM

Kod transformatora koji rade u mreži u kojoj su zvezdišta transformatora uzemljena preko otpora za ograničenje

struja jednopolnog kratkog spoja primena diferencijalne zaštite nije efikasna. Problem je što se može desiti da struje kratkih spojeva budu istog reda ili manje od veličine struje pri kojoj se deluje diferencijalnom zaštitom, pa postoji opasnost da jedan deo kratkih spojeva ne bude detektovan. Iz tog razloga se diferencijalna zaštita primenjuje samo za transformatore većih snaga.
Rad zaštite od kratkih spojeva sa zemljom zasnovan je na detekciji nulte komponente struje.

3.4 TRENUTNA PREKOSTRUJNA ZAŠTITA

Ova zaštita je rezervna ili alternativna diferencijalnoj zaštiti. Ovom zaštitom se eliminišu međufazni kratki spojevi na priključcima transformatora i u transformatoru. Ovom zaštitom je efikasno delovati u slučaju kvarova unutar transformatora ukoliko se pri simulaciji međufaznog kratkog spoja sa minimalnom vrednošću struje na mestu bliže releju dobije vrednost struje koja je bar dva puta veća od struje podešenja.
Ovaj način provere osetljivosti zaštite je pre svega praktičan jer se ne mogu lako izračunati vrednosti struja kada je mesto kvara unutar transformatora.

3.5 TERMIČKA ZAŠTITA

Transformatori su električne mašine koje se mogu preopteretiti. Propisima nije regulisano koliko sme da bude i da traje opterećenje. Zato se u tehničkim preporukama elektroprivrednih preduzeća daju odgovarajuće preporuke za pogone sa preopterećenjima.
Kao zaštita od preopterećenja se kod transformatora SN/NN koristi bimetalna zaštita sa strane niskog napona.
Postupci zaštite od preopterećenja zasnovani na merenju temperature ulja bolja su varijanta zaštite od preopterećenja jer su, posredno, preko temperature ulja uzeti u obzir i prethodno opterećenje i aktuelna temperatura okoline.

3.6 ZAŠTITA OD STRUJA SPOLJNIH KRATKIH SPOJEVA

Pri pojavi kratkih spojeva u blizini transformatora može se dogoditi da kroz transformator teku struje velikog intenziteta koje mogu dovesti do oštećenja. U ovakvim slučajevima termičke zaštite nisu dovoljno efikasne jer reaguju relativno sporo.

Trenutna prekostruja je takođe neosetljiva na ovakve kvarove. Iz tog razloga uvodi se zaštita od struja spoljnih kratkih spojeva.
Ona se realizuje pomoću prekostrujnih releja postavljenih sa jedne ili obe strane tranformatora, u zavisnosti da li postoji jednostrano ili dvostrano napajanje transformatora [1].

4. MIKROPROCESORSKA ZAŠTITA

4.1. ISTORIJSKI RAZVOJ

Razvoj mikroprocesorske zaštite počeo je krajem 60-ih i početkom 70-ih godina. U početku je rad mikroprocesorskih zaštita bio upoređivan sa radom klasičnih releja. Uloženo je mnogo napora da se mikroprocesori učine dovoljno pouzdanim i sigurnim. Prvi komercijalni mikroprocesorski releji su bili frekventni releji.
Cena je nekada bila ograničavajući faktor za značajnije širenje ove zaštite, ali sa značajnim smanjenjem cene čipova i ova prepreka je uklonjena.

4.2 PRINCIP RADA

Ova grupa releja predstavlja savremenu grupu koja se primenjuje u zaštiti. Relativno retko se javlja situacija da se mikroprocesor koristi samo za jednu vrstu zaštite. Najčešći je slučaj da se mikroprocesor koristi kao baza za funkcionalno objedinjavanje više vrsta releja, kao i funkcija koje se sreću u okviru zaštitnog sistema, pa se zato često govori o mikroprocesorskom zaštitnom sistemu ili kraće o mikroprocesorskoj zaštiti.
Mikroprocesorska zaštita podrazumeva zaštitu koja je potpuno numerički orijentisana. Dakle, svi ulazi u deo u kojem se obrađuju merni signali su numeričke vrednosti koje se dobijaju odgovarajućim proračunom. Do potrebnih veličina za proračune se dolazi merenjem (semplovanjem) naponskih i strujnih signala i njihovom konverzijom u binarne vrednosti više puta u nekom periodu vremena. Kod većine releja, broj semplova predstavlja unapred izabrani umnožak nominalne frekvencije koja se koristi u elektroenergerskom sistemu. Pošto se za proračune koriste veličine kao što su npr. Fazori ili efektivne vrednosti, neophodno je dobijene binarne vrednosti transformisati u te veličine. Ove veličine su podaci za proračune nakon kojih se dobija rezultat – vrednost koja se logički poredi sa izabranim podešenjem. Kao rezultat poređenja dobijaju se signali na izlazima koji mogu imati za rezultat delovanje prekidačima, upravljanje, startovanje nekih drugih procedura ili samo obaveštavanje operatera o događajima.

4.3 VRSTE MIKROPROCESORSKIH RELEJA I ZAŠTITA

Postoje sledeće tri grupe mikroprocesorskih releja i zaštita:

- Mikroprocesorski releji čiji se rad zasniva na primeni izuzetno jednostavnih funkcija i kod kojih se ne zahteva velika brzina merenja. Ovi releji su najčešće bazirani na primeni efektivnih vrednosti, a retko na primeni vrednosti kod kojih se mere amplituda i frekvencija.
- Hibridne zaštite kod kojih se brzo merenje obezbeđuje hardverom (npr. DSP čipom). U okviru takvih releja postoji sistem kojim se obezbeđuje odgovarajući nadzor i upravljanje radom jednog ili više mikroprocesora.
- Deljeni releji se dobijaju kada se jedan deo resursa nekog mikroprocesora (a koji se inače koristi za više namena istovremeno) iskoristi i za potrebe relejne zaštite. U ovom slučaju se primenjuju krajnje jednostavne funkcije, a ne može se očekivati ni posebno visok kvalitet zaštite, niti brzina delovanja. Tipično vreme delovanja ovakvih releja je reda veličine 50ms, tako da se ova grupa primenjuje na mestima gde se ne zahteva brza eliminacija kvara. Istovremeno, cena ovih releja je izuzetno niska.

4.4 POREĐENJE MIKROPROCESORSKE ZAŠTITE I KLASIČNE ZAŠTITE

Primena mikroprocesora stvorila je mogućnost za konstruisanjem inteligentnih releja koji se relativno jednostavno i praktično neprekidno prilagođavaju novostečenim iskustvima. Primenom mikroprocesora dobija se zaštita koja ima sledeće osobine:

- Kvalitetnija i znatno širih mogućnosti u odnosu na klasičnu; omogućena je primena „inteligentnih" funkcija i postupaka.

- Veći broj funkcija se povezuje u okviru jedne zaštite.
- Omogućava se nadzor štićenog objekta i u slučajevima kada nema kvara.
- Korišćenje grafičkog (ekranskog) prikaza.
- Detaljan zapis svih parametara kvara.
- Daljinski nadzor i upravljanje.
- Stabilnost rada sa podešenim parametrima.
- Ovakva zaštita služi kao podrška sistemu upravljanja i automatizaciji pogona.
- Omogućava se samonadzor i samokontrola ispravnosti; postoji mogućnost prijave kvara releja i zaštite odmah po nastavku (kod klasičnih releja kvar se može ustanoviti tek pri kontroli ili tek kada relej ne deluje).
- Jednostavnije održavanje i zamena; značajno smanjenje broja ljudi potrebnih za ove poslove; održavanja praktično nema, već se samo nadziru alarmi o eventualnoj neispravnosti; popravke se ne vrše, već samo zamene.
- Značajni smanjen prostor koji se zahteva za montažu releja; objedinjavanjem funkcija releja u mikroprocesorskoj zaštiti gube se žičane veze koje su postojale između klasičnih releja; smanjuje se potreba za kontrolom veza, kao i mogućnost grešaka pri vezivanju i održavanju releja; sa smanjenjem žičanih veza smanjuje se i verovatnoća eventualnog oštećenja tih veza zbog korozije ili oštećenja koje mogu izazvati životinje (npr. glodari).
- Značajno smanjena potrošnja energije.
- Niska cena.

5. PRIMENA MIKROPROCESORSKE ZAŠTITE ZA ZAŠTITU TRANSFORMATORA

5.1 TOPLANA „CERAK“

Kao primer praktične primene zaštite transformatora mikroprocesorskom zaštitom u odgovarajućem softverskom okruženju biće obrađena zaštita transformatora u toplani „Cerak“ (prikazana na slici 5.1.1).

Slika 5.1.1 – Toplana Cerak [2]

5.2 Proračun podešenja zaštite

Konkretan primer podešenja mikroprocesorske zaštite transformatora biće urađen na transformatorskim ćelijama 35 kV postrojenja toplane „Cerak“.

Postrojenje 35 kV se sastoji iz dve dovodne, jedne mernospojne i dve transformatorske ćelije.

Zaštitni uređaj koji je u ovom slučaju primenjujen je SEPAM T87.

Slika 5.2.1 – Izgled zaštitnog uređaja SEPAM T87 [4]

5.2.1 Diferencijalna zaštita

Tabela 5.2.1.1 – Podešenja diferencijalne zaštite [3]

Diferencijalna zaštita (87T)			
Ćelija	Stepen funkcije	Proradna struja[$\times I_{nTR}$]	Vreme reagovanja [s]
Transformatorske ćelije	Stabilisani – $I_d>$	0,3	0
	Nestabilisani – $I_d>>$	12	0

5.2.2 Prekostrujna zaštita – VN strana (51)

Tabela 5.2.2.1 – *Podešenja prekostrujne zaštite [3]*

Prekostrujna zaštita – VN strana (51)		
Ćelija	Proradna struja [A]	Vreme reagovanja [s]
Transformatorske ćelije	160	1

5.2.3 Kratkospojna zaštita – VN strana (50)

Tabela 5.2.3.1 – Podešenja kratkospojne zaštite [3]

Kratkospojna zaštita – VN strana (50)		
Ćelija	Proradna struja [A]	Vreme reagovanja [s]
Transformatorske ćelije	1050	0

5.2.4 Zemljospojna zaštita – VN strana (51N)

Tabela 5.2.4.1 – Podešenja zemljospojne zaštite [3]

Zemljospojna zaštita – VN strana (51N)		
Ćelija	Proradna struja [A]	Vreme reagovanja [s]
Transformatorske ćelije	50	0,5

5.2.5 Zaštita od otkaza prekidača (50BF)

Tabela 5.2.5.1 – Podešenja zaštite od otkaza prekidača [3]

Zaštita od otkaza prekidača (50BF)		
Ćelija	Strujni uslov [A]	Vreme reagovanja [s]
Transformatorske ćelije	50	0,25

5.3 PRIMENA PRORAČUNATIH PODEŠENJA

Za podešavanje zaštitnog uređaja SEPAM T87 koristi se softversko okruženje „SFT2841“.

6. ISPITIVANJE ZAŠTITE

Pored podešenja zaštite, veoma važan korak je svakako i ispitivanje zaštite. Ispitivanje zaštite je obavezan korak pre puštanja u rad u cilju proveravanja podešenih zaštitnih funkcija injektovanjem sekundarnih vrednosti napona i struja i simuliranjem realnih situacija i kvarova. Testiranje pre puštanja u rad najčešće obuhvata:

- Ispitivanje sistema (injektovanje sekundarnih vrednosti baziranih na primarnim parametrima sistema).
- Ispitivanje zaštitnog uređaja (injektovanje sekundarnih vrednosti u cilju provere tačnosti pojedinačnih funkcija zaštite).
- Testiranje logike zaštitnog uređaja.

Osim testiranja pre puštanja u rad, neophodno je i testiranje funkcionalnosti zaštite u sklopu održavanja uređaja. Ovo je potrebno uraditi u cilju održavanja ispravnosti funkcija zaštite Sistema, ali takođe i u cilju produžavanja životnog veka samog uređaja. Prilikom testiranja funkcionalnosti moguće je otklanjanje skrivenih grešaka.

7. UPOTREBA PODATAKA ZA SISTEM UPRAVLJANJA, ZAŠTITE I NADZORA

Sam zaštitni mikroprocesorski uređaj, u zavisnosti od kompleksnosti, raspolaže sa velikim brojem informacija. U cilju isporuke kvalitetne električne energije poželjno je sve informacije pratiti i sagledati u realnom vremenu. U cilju povećanja ekonomičnosti, pouzdanosti, sigurnosti i kvaliteta isporuke električne energije potrebno je povezati transformatorske stanice u sistem pomoću koga bi bilo moguće vršiti daljinsko nadziranje i upravljanje.

SCADA je sistem koji služi za automatizaciju procesa, za prikupljanje podataka sa senzora i dislociranih mernih stanica, za prenos i prikazivanje prikupljenih podataka u centralnoj stanici u svrhu nadzora i/ili upravljanja u realnom vremenu [5, 6].
Za nadgledanje su obično zaduženi operatori, inženjeri i sl. a pod pojmom upravljanja misli se na praćenje, zadavanje pravila rada i telemetriju. Prikupljeni podaci se prikazuju na jednom ili više računara u centralnoj stanici.

8. ZAKLJUČAK

Kvarovi transformatora koji nastaju se eliminišu korišćenjem nekih od zaštita koje su obrađene u ovom radu. Pored ekonomskih razloga (samo transformatori ima visoke cene), razlozi štićenja su svakako i sprečavanje havarijskih dešavanja i ugrožavanje bezbednosti ljudi.
Mikroprocesorska zaštita je svakako prevazišla sve ograničavajuće faktore starih tipova relejne zaštite.
U poslednjim delovima ovog rada prikazane su samo neke od mogućnosti primene mikroprocesorske zaštite (u ovom radu konkretno zaštite transformatora, što je svakako primenjivo i na druge zaštite) koje je omogućio napredak i razvoj tehnologije. Pored pomenutih zaštitnih funkcija, kvalitet zaštite i upravljanja podignut je na viši nivo digitalnim prenosom signala i digitalnom komunikacijom. Ova pomenuta rešenja direktno dovode do prednosti automatizovanih sistema i brojne dobrobiti kao što su: smanjenje troškova održavanja transformatorske stanice u periodu eksploatacije, povećanje pouzdanosti i sigurnosti, smanjenje ljudskih resursa,...

9. LITERATURA

[1] Duško Bekut, „Relejna zaštita“, Novi Sad, Fakultet tehničkih nauka, 2009.
[2] http://www.beoelektrane.rs/
[3] Electroconsult d.o.o „Ispitni protokoli, Electroconsult“, Novi Sad
[4] Schneider Electric „SEPAM mikroprocesorska zaštita“ – srpski korisnički katalog
[5] https://www.scadalink.com/solutions/solutions-by-technology/remote-monitoring-systems/
[6] https://www.ucg.ac.me/skladiste/blog_8554/objava_30673/fajlovi/SCADA%20sistemi2013.pdf

Kratka biografija:

Milica Dukić rođena je u Zrenjaninu 1994. godine. Master rad je odbranila 2019. godine na Fakultetu tehničkih nauka iz Elektrotehnike i računarstva – Elektroenergetski sistemi.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/07BE22Ubiparip

PREGLED HVDC PRENOSA

A REVIEW OF HVDC TRANSMISSION

Miljan Ubiparip, Darko Marčetić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Visokonaponski prenos jednosmjernom strujom (High Voltage Direct Current - HVDC) je bio alternativna metoda u prenosu električne snage sa jedne lokacije na drugu. Razvojem energetske elektronike otvaraju se nove primjene HVDC prenosa. U ovom radu su razmatrane bitne osobine HVDC prenosa, kao i primjena energetske elektronike u sistemima baziranim na ovoj tehnologiji. Naglašene su prednosti, kao i nedostaci u odnosu na naizmjenične sisteme. Na kraju će biti dat ostvrt na DC prenosne mreže.*

Ključne reči: *HVDC prenos, LCC konvertor, VSC konvertor, MMC konvertor, DC prenosna mreža*

Abstract – *High Voltage Direct Current (HVDC) transmission has been an alternative method of transmitting electric power from one location to another. The development of power electronics creates various applications of HVDC transmission. Important characteristics of HVDC transmission and application of power electronics in systems based on this technology are discussed in this paper. Advantages, as well as disadvantages comparing to AC systems are emphasized. DC transmission grids will be reviewed at the end of this paper.*

Keywords: *HVDC transmission, LCC converter, VSC converter, MMC converter, DC transmission grid*

1. UVOD

Dugo vremena se elektroprivreda doživljavala kao stabilan i potpuno sazreo sektor, koji dobro posluje bez puno inovacija. Poslednjih decenija situacija se drastično promenila.

Potreba za energijom je sve veća i jedno od rješenja je prenos energije iz izvora električne energije koji su geografski udaljeni u odnosu na mjesta potrošnje. Ali, sa uvećanjem dužine prenosa, problemi u AC prenosu energije se višestruko usložnjavaju. Nadalje, ulaskom obnovljivih izvora nacionalni EES postaju sve nestabilniji, i javlja se potreba da se ti međusobno veoma udaljeni sistemi povremeno pomažu razmjenom energije.

Konačno, da bi se u datom trenutku vremena maksimalno iskoristila energija svih raspoloživih obnovljivih izvora, optimalno rešenje bi bila globalna elektroenergetska mreža koja bi omogućila povezivanje i razmenu energije između geografski udaljenih distribuiranih izvora i potrošača.

NAPOMENA:

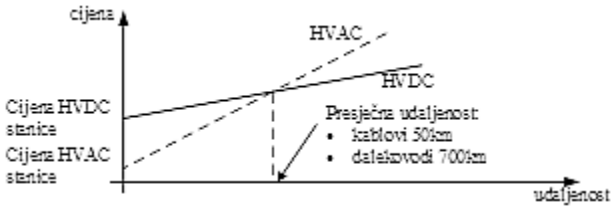
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Darko Marčetić, red. prof.

Iz gore navedenih razloga razvijen je visokonaponski prenos jednosmjernom strujom (High Voltage Direct Current - HVDC) i dokazano je da je on najbolje rješenje za mnoga područja primjene, a broj instalacija po godini neprestano raste od početka dvadeset prvog veka. Unapređenje uređaja energetske elektronike (najviše IGBT tranzistora) u poslednje dve decenije je izmjenio odnos prema HVDC prenosu.

2. POREĐENJE AC I DC PRENOSA

Usvajajući da su karakteristike izolatora za AC i DC prenos slične, DC vod može prenijeti istu snagu sa dva provodnika kao što to može AC vod sa tri provodnika iste veličine. Zbog navedenog razloga, DC vodovi zauzimaju mnogo manje zemljišta, jednostavnije i jeftinije stubove, kao i manje troškove provodnika i izolatora.

Sa aspekta gubitaka snage, zbog korišćenja samo dva provodnika, DC prenos ima za dve trećine manje gubitke nego tradicionalni AC prenos. Pored ovoga, pri DC prenosu, nema se površinskog efekta i dielektrični gubici su mnogo manji, pa imamo smanjenje gubitaka u odnosu na AC prenos. Ostali značajni troškovi prenosa jesu troškovi prekidačkih (ispravljačkih i invertorskih) stanica u slučaju DC prenosa. Usljed prisustva elektro-energetskih pretvarača i filtera, cena izrade HVDC stanice je značajno veća nego cena izrade tradicionalne HVAC stanice. Na slici 1. prikazano je poređenje cijene HVDC i HVAC prenosa u zavisnosti od dužine dalekovoda/kablova.



Slika 1. *Poređenje cijene HVDC i HVAC za nadzemne vodove i za kablove*

Zahvaljujući svojoj brzoj kontroli, prednosti DC prenosa u odnosu na AC prenos su potpuna kontrola nad prenešenom snagom, poboljšane prelazna i dinamička stabilnost u pridruženim AC mrežama, brza kontrola ograničavanja struje kratkog spoja u DC linijama.

3. TIPOVI HVDC SISTEMA

U zavisnosti od lokacije, tipa, svrhe upotrebe, kao i izbora kabla, postoji par različitih konfiguracija HVDC prenosa:

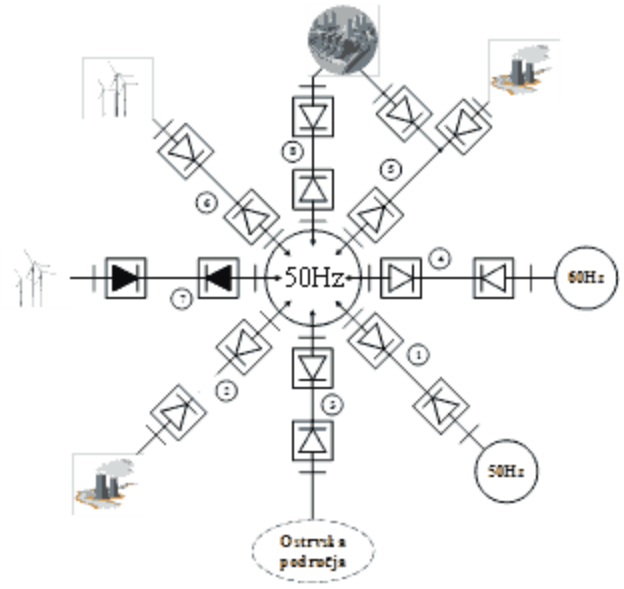
- Monopolarni,
- Bilolarni,
- Back-to-back,
- Multiterminalni i
- Tripolarni HVDC prenos.

Monopolarna konfiguracija HVDC prenosa se sastoji od dvije konvertorske stanice i DC linije koja ih povezuje (jedan provodnik). Bipolarna konfiguracija HVDC prenosa predstavlja paralelnu vezu dva monopolarna sistema. U slučaju kvara na jednoj od dvije linije, druga nastavlja da radi po principu monopolarne konfiguracije, koristeći zemlju kao povratni vod.
Promjena smjera toka snage je moguća, jer u slučaju potrebe, jedna pretvaračka stanica prelazi iz rada ispravljake u rad invertorske stanice, a druga obrnuto. Back-to-Back HVDC prenosni sistem se sastoji od dvije pretvaračke stanice postavljene na istoj lokaciji. Kod ove konfiguracije, ne postoji DC linija prenosa. Ova konfiguracija koristi se za povezivanje dvije AC mreže. Multi-terminalna konfiguracija HVDC prenosnog sistema se sastoji od DC prenosne linije i više konvertora, koji mogu biti povezani redno ili paralelno.
Na ovaj način, pruža se dodatna pouzdanost sistema, jer se za cilj ovakve konfiguracije ima povezivanje više pojedinačnih AC sistema u jedan zajednički sistem. Tripolarna konfiguracija HVDC prenosnog sistema je najnovija konfiguracija HVDC prenosa. Koristi se u sistemima baziranim na MMC konvertorima.

4. PRIMJENA HVDC PRENOSA

Na slici 2. prikazane su razne primjene HVDC prenosnog sistema:

1. Prenos snage dalekovodima na veliku daljinu. Pri ovoj primjeni, HVDC sistem postaje isplativ za udaljenosti preko 700km,
2. Prenos snage podvodnim kablovima na veliku daljinu. Usljed problema velike kapacitivnosti i problema sa kompenzacijom reaktivne snage, kod ovakve primjene HVDC prenos postaje isplativ već nakon 50km udaljenosti,
3. Stabilizacija protoka električne energije u integrisanom elektroenergetskom sistemu. Strateški postavljen HVDC sistem može prevazići problem prigušenja elektromagnetskih oscilacija i stabilnosti mreže. Zbog brze kontrole, HVDC sistem pruža prijeko potrebna prigušivanja i pravovremene mogućnosti preopterećenja,
4. Povezivanje dvije AC mreže različitih frekvencija, kao i dvije nesinhronizovane AC mreže. Bez obzira na razliku u frekvenciji ili faznoj razlici, dvije AC mreže mogu biti povezane koristeći HVDC sistem (osim u slučaju kada je fazna razlika velika, tada nije moguća direktna veza),
5. Kada se snaga treba prenositi sa udaljene lokacije kroz različite države ili različita područja unutar jedne države, moguće je koristiti multiterminalni HVDC prenos,
6. U slučajevima kada se koriste obnovljivi izvori električne energije koji su obično na udaljenim lokacijama od potrošača,
7. Sve veći značaj u HVDC sistemima zauzimaju sistemi sa naponskim pretvaračima,
8. Budući da se reaktivna snaga ne prenosi preko DC linije, dva AC sistema se mogu povezati preko HVDC linije bez povećanja snage kratkog spoja.



Slika 2. *Oblasti primjene HVDC prenosa*

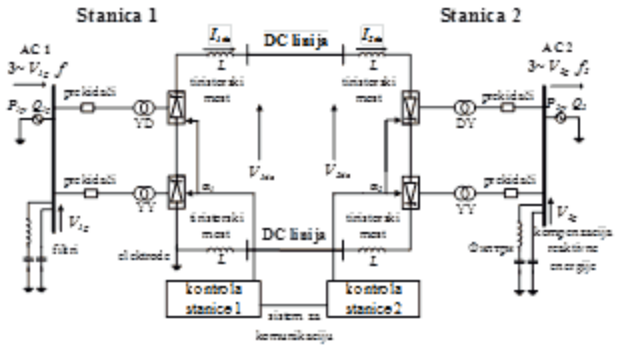
5. ENERGETSKA ELEKTRONIKA U HVDC PRENOSU

Da bi konverzija električne energije iz naizmjeničnih sistema u jednosmjerne bila moguća, neophodno je postojanje energetskih pretvarača električne energije. Dvije osnovne konfiguracije HVDC sistema potrebne da bi ovaj proces konverzacije bio ostvaren su:

- HVDC sa strujnim konvertorima (Line-Commutated Converter - LCC) i
- HVDC sa naponskim konvertorima (Voltage Source Converter - VSC).

5.1 HVDC sa strujnim konvertorima

HVDC se sastoji iz sistema za prenos i konvertora. Tipičan dijagram linijski komutovanog HVDC je dat na slici 3.



Slika 3. *Blok dijagram linijski komutovanog HVDC sistema*

Osnovni dijelovi ovog sistema su:

- Pretvarači su jedan ili više 6-pulsnih tiristorskih mostova povezanih serijski u 12 ili 24 pulsnoj konfiguraciji. Svaki most u sebi poseduje na stotine tiristora,
- Specijalni transformatori. Osnovna osobina im je da rade sa strujama i naponima bogatog harmonijskog sastava i da mogu da izdrže naponske i strujne udare.
- Prigušnice sa DC strane reda veličine 0.5H. Njihova uloga je da uvećaju dinamičku stabilnost, smanje struju greške i priguše komutacione harmonike,

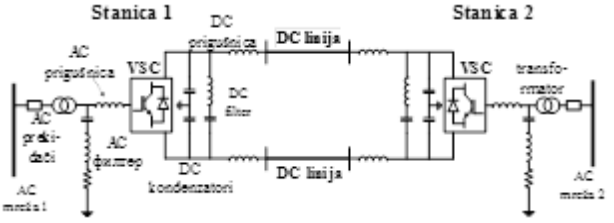
• Kompenzatori reaktivne snage. HVDC konvertori zahtevaju i do 60% reaktivne snage u odnosu na nominalnu snagu. Iz tog razloga se postaju filterske banke koje daju veliku količinu te snage i time se ne opterećuje AC sistem. Banke za kompenzaciju reaktivne snage su varijabilne, tj menjaju se u zavisnosti od količine prenesene energije koja definiše količinu potrebne reaktivne snage.

• Filtri. Tipičan 12-tiristorski pretvarač utiskuje jedanaesti, trinaesti, dvadeset treći i dvadeset peti harmonik u AC mrežu pa se sa AC strane postavljaju filtri na tim frekvencijama. Ponekad se filtri stavljaju i na DC stranu.

Kontrola tiristorskog ispravljača ima dva osnovna cilja, da utisne željenu DC snagu u HVDC liniju i da obezbedi sinhronizaciju sa mrežom. Kontroler se sastoji iz unutrašnje strujne regulacione konture koja upravljanjem uglom ukljućenja tiristora reguliše struju na željenom nivou i spoljašnje regulacione kontrole snage koja upravljanjem snagom reguliše snagu utisnutu u HVDC na zadatom nivou. Sinhronizacija sa mrežom se zasniva na klasičnoj PLL strukturi koja prati ugao mreže i obezbeđuje da je ugao uključenja sinhronizovan sa prolascima mrežnog napona kroz nulu. Kontrola snage koja se utisne u HVDC prenosnu liniju je primarna funkcija HVDC ispravljača.

5.2 HVDC sa naponskim konvertorima

Naponski konvertori (Voltage Source Converter – VSC) koriste IGBT prekidače koji su samo komutujući, tj. uključuju se i isključuju u zavisnosti od kontrolnog signala na kontrolnoj elektrodi (gate), nezavisno od napona mreže. Ovim je omogućeno PWM upravljanje koje generiše više prekidačke frekvencije (1kHz) i samim tim su oblici izlaznih DC i AC napona mnogo približniji željenim, sa manjim varijacijama napona i struje. Ove varijacije su ujedno i na višim učestanostima, tako da su potrebni manji i jeftiniji filtri na izlazima VSC pretvarača. VSC se mogu u potpunosti sinhronizovati sa mrežom, i time nezavisno upravljati aktivnom i reaktivnom snagom. VSC neosporno obezbeđuju kvalitetniji HVDC prenos. Jedini problem je cijena, koja ograničava snagu na kojoj ovaj tip prenosa može da se primeni.



Slika 4. *Blok šema HVDC sa dva naponska konvertora sa IGBT prekidačima*

Osnovni dijelovi VSC HVDC pretvaračke stanice su:

• AC prekidači. AC prekidači se koriste za uključivanje/isključivanje sistema tokom normalnih i havarijskih stanja. Sa aspekta dizajna, ne postoje posebni zahtjevi u odnosu na tradicionalne sisteme napajanja,

• Transformator. Koristi se trofazni transformator sa kontrolom regulacione preklopke (osim u slučaju MMC HVDC sistema, kada se mogu kloristiti standardni transformatori),

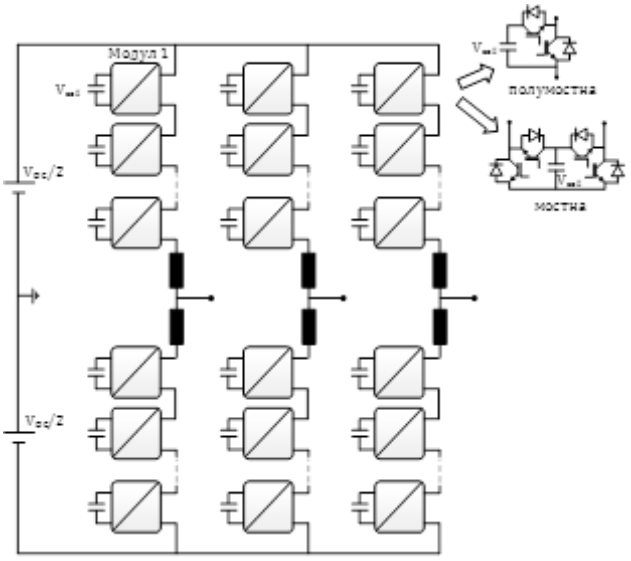
• AC filtri. Ovi filtri imaju niže nazivne vrijednosti od onih korišćenih u LCC HVDC konfiguraciji i nije potrebno da obezbjede kompenzaciju reaktivne snage. Niskopropusni LC filter se obično koristi za suzbijanje viših harmonika i sprječavanje interakcije sa osnovnim harmonicima,

• DC kondenzatori. Ovo uređaji služe za skladištenje energije u VSC konfiguraciji kao i filtraciju harmonika.

• DC filtri. Koriste se umjesto povećavanja DC kondenzatora, za eliminaciju određenih harmonika.

• Pretvarač koji se sastoji od niza modula (ćelija) koji će biti detaljnije objašnjeni u nastavku.

5.3 Modularni naponski konvertor – MMC

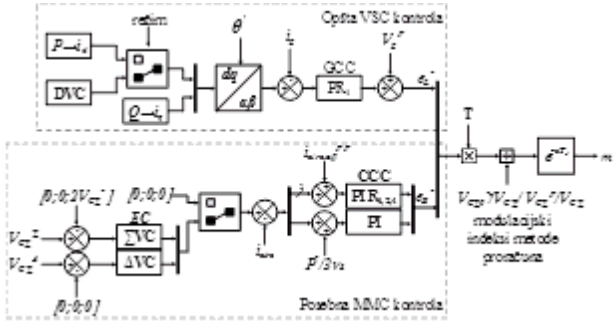


Slika 5. *Trofazni MMC*

Osnovna ideja modularnih konvertora sa više nivoa (Multilevel Modular Converter – MMC) je da se jedan isti prekidački modul koristi u svim topologijama. MMC su veoma atraktivni jer kada se napravi jedan dobar modul onda se topologije, broj nivoa, broj grana, razlikuju samo po načinu i broju povezanih modula. MMC su danas dominantna tehnologija u HVDC i u drugim visokonaponskim primjenama energetske elektronike.

Osnovna gradivna jedinica MMC konvertora je modul, ili ćelija. Postoje dvije vrste modula, polumostni sa dva prekidača, i mostni sa četiri prekidača.

Upravljačka struktura trofaznih VSC konvertorom sa MMC topologijom je data na slici 6. (invertorski režim rada). Konvertor se sinhronuzuje sa mrežom na koju je povezan preko klasične PLL strukture. U tom slučaju koordinatni sistem sinhrono rotira sa naponskim vektorom mreže i aktivnu snagu je moguće regulisati upravljanjem d komponentom mrežne struje. Reaktivna snaga se reguliše nezavisno, upravljanjem q komponentom mrežne struje. Ovaj dio upravljačke strukture na slici je obilježen kao „Opšta VSC kontrola“. Razlika u upravljanju MMC VSC se javlja u dijelu koji se bavi kontrolom balansa plivajućih kondenzatora (slika 6. dio „Posebna MMC kontrola“).



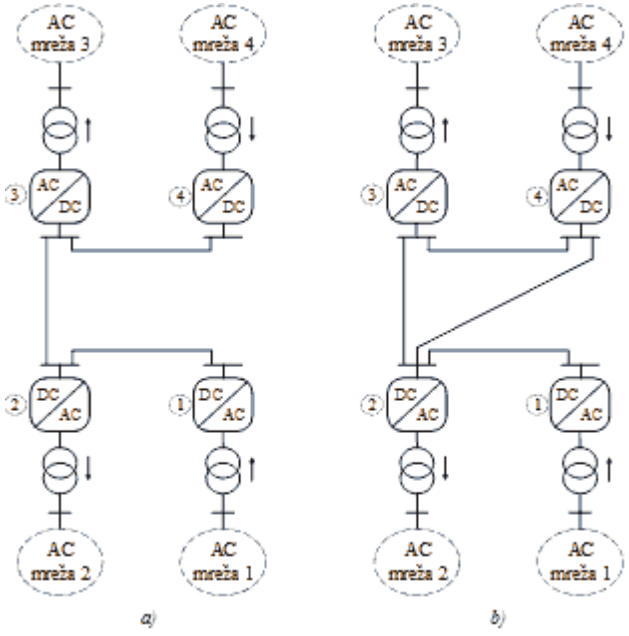
Slika 6. *Pojednostavljeno upravljanje MMC konvertorom u invertorskom režimu rada*

6. DC PRENOSNE MREŽE

Osnovna funkcija DC mreže jeste povezivanje HVDC linkova i omogućavanje razmjene snage između svih DC terminala. Ovo donosi mnogobrojne prednosti kao što su: bolje iskorišćenje prostora (zemljišta), veću pouzdanost i sigurnost prenosa, veću efikasnost, bolju razmjenu snage, veću operativnu fleksibilnost i sve ostale pogodnosti koje nudi međusobno uvezivanje sistema. DC mreža pruža pristup svim učesnicima na tržištu, što uključuje obnovljive izvore električne energije koji su od izuzetnog značaja za sadašnje i buduće planove izrade prenosnih mreža.

Arhitektura DC mreže može biti serijska ili paralelna, Pošto serijska arhitektura ima brojne nedostatke, neće biti obrađena u ovom radu.

U paralelnoj arhitekturi DC mreže, pretvaračke stanice su spojene paralelno, te stoga rade sa istim naponom. DC mreža može biti radijalna ili upetljana.



Slika 7. *Paralelna arhitektura: a) radijalne DC mreže, b) upetljane DC mreže*

7. ZAKLJUČAK

Visokonaponski prenos jednosmjernom strujom je sigurna i efikasna tehnologija namjenjena za prenos velike količine električne energije na velike udaljenosti. Zbog svojih mnogobrojnih prednosti u odnosu na tradicionalne naizmjenične prenosne sisteme, HVDC sistemi veoma brzo rastu i postaju važan dio prenosnih mreža. Upotrebom IGBT tranzistora u HVDC tehnologiji počinje novo poglavlje u njenom razvoju. Dolazi se do HVDC tehnologije sa naponskim konvertorima, čime se povećava fleksibilnost ove tehnologije i njene primjene u prenosu električne energije. Konačno, pojava modularnih konvertora sa više nivoa ispostavila se kao najisplativiji koncept naponskih konvertora i kao takva postala izuzetno značajan aspekt daljeg razvoja HVDC tehnologije. Nadalje, usljed sve veće potrošnje električne energije i sve veće raznovrsnosti pri proizvodnji iste, javlja se potreba za međusobnim uvezivanjem HVDC sistema u mreže, koje pretenduju da u budućnosti budu globalnih razmjera.

8. LITERATURA

[1] Dragan Jovcic, Khaled Ahmed,"*High Voltage Direct Current Transmission: Converters, Systems and DC Grids*", John Wiley and Sons Ltd, United States, 2015.

[2] Chan-Ki Kim, Vijay K. Sood, Gil-Soo Jang, Seong-Joo Lim and Seok-Jin Lee,"*HVDC Transmission: Power Conversion Applications in Power Systems"*, John Wiley & Sons (Asia), Singapur, 2009.

[3] Vijay K. Sood,"*HVDC and FACTS Controllers Applications of Static Converters in Power Systems*", Kluwer Academic Publishers, Boston, 2004.

[4] Nemanja Savić, Vladimir Katić, "*Overview of the Configuration and PowerConverters in High Voltage Direct Current Transmission Systems*", Kladovo, Srbija, 2017.

[5] Dražen Dujić," *MMC-BASED CONVERSION FOR MVDC APPLICATIONS*", Indel, Banja Luka, 2018.

[6] Nilanjan Ray Chaudhuri, Balarko Chaudhuri, Rajat Majumder, Amirnaser Yazdani "*MULTI-TERMINAL DIRECT-CURRENT GRIDS Modeling, Analysis, and Control*", John Wiley & Sons, Inc., Hoboken, New Jersey, 2014.

Kratka biografija:

Miljan Ubiparip rođen je u Stocu, BiH, 1991. godine. Srednju tehničku školu završava u Trebinju. 2017. godine završava osnovne akademske studije na Fakultetu tehničkih nauka u Novom Sadu na studijskom programu Energetika, elektronika i telekomunikacije. Nakon toga upisuje master akademske studije, usmjerenje Elektroenergetika – energetska elektronika i električne mašine.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/07BE23Simic

LOKACIJA KVARA U DISTRIBUTIVNIM MREŽAMA

FAULT LOCATION IN DISTRIBUTION NETWORKS

Aleksa Simić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Ovaj rad se bavi problematikom lokacije kvara u distributivnim mrežama. Dat je pregled mogućih metoda za lokaciju kvara. Obrađene su impedantne metode kao i metode bazirane na merenju propada napona. Na kraju rada izvršeno je testiranje strujne i impedantne metode na primeru test ditributivne mreže.*

Ključne reči: *Distributivna mreža, Lokacija kvara*

Abstract – *This paper deals with the problem of fault location in distribution networks. An overview of possible methods for fault location is given. Impedance methods as well as methods based on the measurement of voltage drop are discussed. At the end of the paper, the current and impedance methods were tested using the distribution test network.*

Keywords: *Distribution network, Fault location*

1. UVOD

Elektroenergetski sistemi su konstantno izloženi kvarovima, što utiče na njihovu pouzdanost, sigurnost i kvalitet električne energije. Najveći broj kvarova se dešava u distributivnoj mreži, pri čemu su najčešći kvarovi kratki spojevi. Prekid napajanja potrošača je nepoželjno stanje, te je potrebno preduzeti odgovarajuće mere da se ono što pre eliminiše. Određivanje mesta kvara u distributivnim mrežama je od presudnog značaja, jer se na taj način smanjuje dužina prekida napajanja potrošača. U drugoj glavi dat je pregled tipova kratkih spojeva, predložene su mere za smanjenje broja kvarova, prezentovane su nove mogućnosti vizuelne inspekcije nadzemnih vodova. U trećoj glavi dat je osnvni koncept pametnih mreža kao i sistema za upravljanje prekidima napajanja. Razmatran je uticaj ugradnje pametnih brojila na unapređenje ovog sistema. Takođe opisani su i uređaji distributivne automatike. Četvrta glava bavi se raznim tehnikama i algoritmima za lokaciju kvara. U petoj glavi obrađena je strujna i impedantna metoda za lokaciju kvara čije je testiranje izvršeno u šestoj glavi. U sedmoj glavi iznesen je zaključak, dok osma glava daje spisak literature.

2. KVAROVI U DISTRIBUTIVNIM MREŽAMA

Kvar predstavlja poremećeno stanje. Najčešći tip kvarova predstavljaju kratki spojevi. Kratak spoj je slučajan ili nameran spoj provodnika, preko malog otpora ili impedanse, između dve ili više tačaka, koje su na različitim naponima [1].

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je dr Duško Bekut, redovni profesor.

Razlikuju se sledeći tipovi kratkih spojeva: jednopolni kratak spoj, dvopolni kratak spoj, dvopolni kratak spoj sa zemljom, tropolni kratak spoj, tropolni kratak spoj sa zemljom.

3. AUTOMATIZACIJA DISTRIBUTIVNIH MREŽA

Korišćenje naprednih algoritama za lokaciju kvara moguće je jedino ako je distributivna mreža automatizovana u dovoljnoj meri. To podrazumeva instaliranje brzih mernih jedinica, snimača kvara, indikatora kvara kao i reklozera i sekcionalizera.

U grupu snimača kvara spadaju uređaji koji imaju sposobnost da brzo registruju trenutne vrednosti napona i struje [2]. Dva glavna tipa snimača kvara koji se najčešće koriste pri proceni mesta kvara u mrežama su digitalni snimači kvara i mikroprocesorski releji. Pri upotrebi snimača kvara za procenu mesta kvara se imaju sledeći podaci: fazori napona i struja u sve tri faze u polju transformatora sa niskonaponske strane i izvod na kojem se kvar desio. Na osnovu vrednosti napona i struje prvo se identifikuje tip kvara, a zatim se izračunava (određuje) moduo merene struje kvara kao i vrednost merene impedanse na mestu, gde je snimač kvara instaliran (na mestu merenja). Ta impedansa uz podatak na kojem se izvodu desio kvar predstavlja osnovu za procenu rastojanja do mesta kvara.

Indikatori prolaska struje kvara su uređaji koji se postavljaju na strateškim mestima da bi obezbedili informacije o prolasku struje kvara. U radijalnim distributivnim mrežama bez distribuiranih generatora koriste se neusmereni indikatori prolaska struje kvara. Ukoliko indikator detektuje struju kvara, to znači da se kvar desio iza indikatora. Indikatori kvarova mogu biti dizajnirani da informaciju o struji kvara prikazuju lokalno ili da je šalju operateru u SCADA (*Supervisory Control and Data Acquisition*) system [3].

3.1 OMS (*Outage Management System*)

OMS predstavlja deo DMS-a (*Distribution Management System*) i moćan je alat za brzo i efikasno otklanjanje beznaponskih stanja u mreži. On pomaže dispečeru da na najoptimalniji način obavi sve korake i angažuje dostupne resurse koji su neophodni za ponovno vraćanje mreže u stanje pre kvara. OMS treba obezbediti automatizaciju koraka ili čitave akcije dispečera, ekipa na terenu i administracije vezane za izdavanje radnih naloga, tako da se vreme potrebno za vraćanje u normalno stanje mreže znatno skrati. Na taj način OMS igra ključnu ulogu u svakodnevnom vođenju mreže i smanjenju broja i trajanja prekida napajanja.

OMS je moguće unaprediti integracijom sa AMI sistemom (*Advanced Metering Infrastructure*) tj. ugradnjom pametnih brojila na mestu isporuke električne energije. Pametna brojila mogu automatski ili na zahtev pružati informacije o statusu uključenosti. Automaski generisane informacije podrazumevaju indikaciju nestanka napajanja kada je nestalo struje i indikaciju kada se napajanje ponovo uspostavi. Nakon isključenja, brojilo i dalje ima dovoljno rezervnog napajanja kako bi poslalo poruku centralnoj jedinici da je napajanje prekinuto na mestu ugradnje tog uređaja.

4. METODE LOKACIJE KVARA U DISTRIBUTIVNIM MREŽAMA

Lokacija kvara predstavlja izazov u distributivnim mrežama zbog njihove topologije i specifičnih operativnih karakteristika, kao i nedostatka raspoloživih merenja. Još uvek se često koriste tehnike vizuelne inspekcije za procenu mesta kvara. Takve tehnike nisu pogodne za kablovske sisteme, i zahtevaju dugo vreme za pronalazak lokacije kvara u većim distributivnim mrežama.

Automatska lokacija kvarova se najčešće koristi. Zasniva se na određivanju fizičkog mesta kvara pomoću obrade vrednosti naponskog i strujnog talasa.

Sledeći faktori utiču na procenu mesta kvara u distributivnoj mreži: promena konfiguracije mreže, razrgranati vodovi, nehomogeni vodovi, otpor na mestu kvara, opterećenje priključeno duž izvoda, prisustvo distributivnih generatora kao i način uzemljenja neutralne tačke.

4.1 Impedante metode za lokaciju kvara

Korišćenje napona i struja na terminalima, zajedno sa parametrima vodova, predstavlja najjednostavniji način utvrđivanja mesta kvara. Pretpostavlja se da je izračunata impedansa deonice pod kvarom merilo razdaljine do mesta kvara. Metode koje spadaju u ovu katergoriju nazivaju se impedantne metode za lociranje kvara. One su ekonomične i jednostavne za primenu. U zavisnosti od ulaznih signala lokatora kvarova, ove metode se mogu dalje klasifikovati. U zavisnosti od toga da li se koriste merenja sa jednog ili oba kraja voda, ove metode se klasifikuju na sledeći način:

- metode jednog kraja,
- metode oba kraja.

4.2 Metoda zasnovana na simulaciji podataka o propadu napona

Ova metoda predlaže pronalaženje lokacije kvara poređenjem izmerene vrednosti propada napona sa vrednostima dobijenim simulacijom i pohranjenim u bazu podataka. Merenje se vrši na početku izvoda. Uvažavajući činjenicu da deonice distributivnih izvoda nisu homogene (različitih su materijala i poprečnih preseka), potrebno je za svaku deonicu napraviti zasebnu jednačinu koja opisuje promenu napona pri kratkom spoju u funkciji dužine deonice. Minimalno bi trebalo izvršiti simulacije na početku i kraju svake deonice i pritom zabeležiti vrednost amplitude i faznog stava napona. Jednačine za svaku krivu modeluju se polinomom drugog stepena. Generalne forme jednačina mogu se predstaviti preko sledećih izraza [3]:

$$V_i = a_0 d^2 + a_1 d + a_2, \quad (4.1)$$

$$\theta_i = b_0 d^2 + b_1 d + b_2. \quad (4.2)$$

Ove jednačine odgovaraju *i*-toj sekcji.

Algoritam poređenja podataka se sastoji od tri koraka [4]:

1) Globalna pretraga.
2) Estimacija udaljenosti kvara.
3) Rangiranje sekcije u kvaru.

U prvom koraku algoritma cilj je odrediti sve potencijane sekcije sa kratkim spojem. Sekcija je selektovana ako amplituda i fazni stav napona upadaju u granice definisane za tu sekciju. Recimo, da bi sekcija *i-j*, gde su *i* i *j* početni i krajni čvor sekcije, bila okarakterisana kao potencijalna sekcija sa kvarom potrebno je da budu ispunjeni sledeći uslovi [4]:

$$V_i \leq V^m \leq V_j, \quad (4.3)$$

$$\theta_i \leq \theta^m \leq \theta_j. \quad (4.4)$$

U drugom koraku estimira se udaljenost mesta kvara od početnog čvora sekcije (čvor *i*). Ova udaljenost proračunava se na sledeći način [3]:

$$d_{F1} = \left(-a_1 \pm \sqrt{a_1^2 - 4a_2(a_0 - V^m)}\right)/2a_2, \quad (4.5)$$

$$d_{F2} = \left(-b_1 \pm \sqrt{b_1^2 - 4b_2(b_0 - \theta^m)}\right)/2b_2 \quad (4.6)$$

$$d_F = (d_{F1} + d_{F2})/2. \quad (4.7)$$

U slučaju da postoji više potencijalnih deonica sa kvarom potrebno je utvrditi njihov prioritet.

4.1 Metoda zasnovana na korišćenju inteligentnih mernih instrumenata

Korišćenje inteligentnih mernih uređaja (*smart feeder meters*) instaliranih na raznim mestima u srednjenaponskoj distributivnoj mreži, otvara nove mogućnosti za rešenje problema lokacije kvara. Ovi uređaji imaju mogućnost prijave kvara kao i merenja napona sa klasom tačnosti od 0.1 [%] do 0.5 [%].

Glavna ideja ove metode jeste korišćenje informacija o vrednosti propada napona kako bi se proračunali ideksi lokacije kvara koji ukazuju na čvor koji je nalbliži mestu kvara. Propad napona (ΔV) računa se na osnovu sledećeg izraza [5]:

$$\Delta V_i^{(abc)} = V_i^{(abc)p} - V_i^{(abc)k}, \quad (4.8)$$

gde su $V_i^{(abc)p}$ i $V_i^{(abc)k}$ amplitude napona u fazama *a,b* i *c* izmerenog u čvoru *i* pre i posle kvara.

Ukoliko je poznata matrica admitansi moguće je proračunati vrednost indeksa δ_k za svaki čvor u mreži. Za čvor sa najmanjom vrednošću ovog indeksa smatra se da je najbliži mestu kvara. U pojedinim slučajevima može se desiti da više čvorova ima istu vrednost indeksa kvara što implicira višestruku estimaciju kvara. Ovaj problem može se rešiti automatskim mapiranjem otkaza koji se sužava oblast pretrage. Na osnovu prorade reklozera i ostalih zaštitnih uređaja, pretraga se ograničava samo na deenergizovani deo mreže tj. obrađuju se podaci samo sa onih uređaja koji su ugrađeni na tom delu mreže [5].

5. LOKACIJA KVARA U DMS SISTEMU

Postoje sledeće metode za lokaciju kvara:

- Strujna metoda,
- Impedantna metoda.

5.1 Strujni metod lokacije kvara

Osnovna ideja ovog algoritma počiva na tome da se u prvom koraku na osnovu podataka o kvaru (prorada određenih releja kao i merenja vrednosti struja u sve tri faze) koji su prikupljeni iz realne mreže identifikuje izvod i tip kratkog spoja. U drugom koraku se pomoću matematičkog modela distributivne mreže simulira identifikovani tip kratkog spoja duž datog izvoda sa ciljem da se odredi mesto na izvodu gde će za zadati kvar dobiti vrednost struje koja je merena [6].

Procena mesta kvara strujnim algoritmom započinje sa identifikacijom izvoda na kome se dogodio kvar. Izvod na kome se dogodio kvar određuje se na osnovu prorade relejne zaštite koja je pozicionirana na izvodnom polju.

Tip kvara se određuje na osnovu rezultata merenja, tako što se identifikuju faze u kojima je povećana struja (snižen napon).

Zatim se za tip kvara koji je prethodno identifikovan simulira na sabirnicamasa kojih polazi izvod sa kvarom a zatim i na sabirnicama na krajevima svih deonica tog izvoda.

Za svako mesto kvra izračunava se vrednost struje koja bi se merila brzom mernom jedincom. Na taj način kada se za svaku deonicu dobije određeni par struja, i ako je merena struja baš u tom intervalu onda je kvar baš na toj deonici.

Izrazom 5.1 definiše se potreban i dovoljan uslov lokacije kvara strujnom metodom [5]:

$$j_i^k \leq j^{mer} \leq j_i^p, \tag{5.1}$$

gde su j_i^k i j_i^p intenziteti struja na kraju i početku deonice *i*.

5.2 Impedantni metod lokacije kvara

Ova metoda se zasniva na principima koji se primenjuju kod distantne zaštite. Na osnovu vrednosti struje i napona koja su zabeležena mernim uređajem u SN polju napojnog transformatora VN/SN, računa se impedansa između merne jedinice i mesta kvara.

Na osnovu prorade releja se zna na kom izvodu se desio kvar, a na osnovu merenja se dobija i tip kvara, tako što se identifikuju faze u kojima je povećana struja (odnosno, snižen napon) [6].

Kao i kod svih distantnih releja, za meru udaljenosti uzima se pogonska impedansa voda do mesta kvara (ta impedansa odgovara impedansi za direktni režim). Izbor impedanse direktog redosleda je posledica činjenice da direktni režim postoji kod svih kvarova.

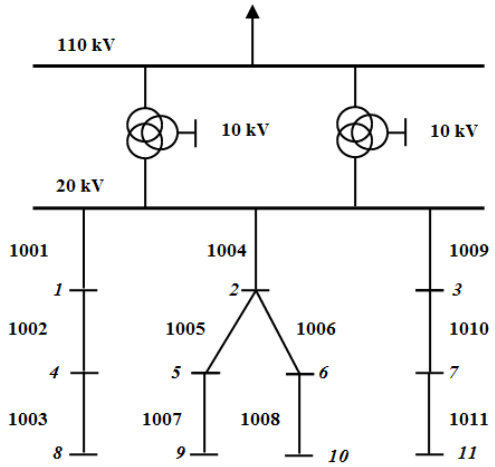
Skup primarnih deonica sa kvarom se identifikuje pomoću sledeće relacije [6]:

$$Im\{Z_{ff1}\} \leq Im\{Z_{ff}\} \leq Im\{Z_{ff2}\}, \tag{5.2}$$

gde su Z_{ff1} i Z_{ff2} impedanse deonice na početku i kraju, a sa Z_{ff} je označena izračunata impedansa.

6. PROVERA RADA STRUJNE I IMPEDANTNE METODE NA TEST PRIMERU

Slika 6.1. predstavlja distributivnu mrežu nad kojom je vršeno testiranje. Bitno je naglasiti da su napojni transformatori 110/20/10 [kV]/[kV]/[kV] uzemljeni preko zajedničkog otpornika otpornosti 40 [Ω].



Slika 6.1. – Jednostavna distributivna mreža

6.1 Lokacija kvara strujnom metodom

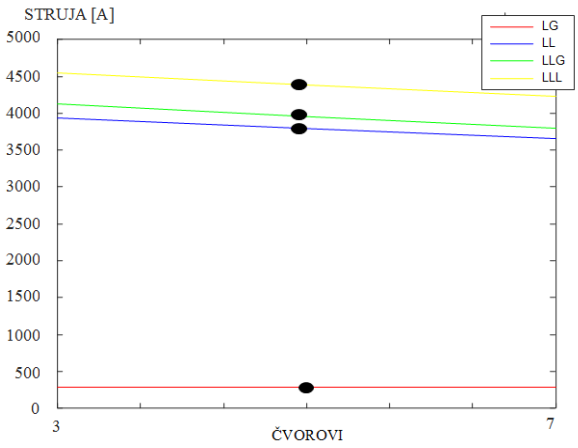
U tabeli su dati rezultati proračuna lokacije kvara primenom strujne metode.

Tabela 6.1.1. – Rezultati proračuna lokacije kvara strujnom metodom

Tip kratkog spoja	Merena struja kvara [A]	Struja kvara čvora 3 [A]	Struja kvara čvora 7 [A]	Grana
1PKS	279.850	281.10	278.61	1010
2PKS	3794.53	3936.13	3660.23	1010
2PKSZ	3858.49	4001.21	3723.06	1010
3PKS	4381.55	4545.05	4226.47	1010

Na slici su prikazani graficki rezultati proračuna lokacije kvara strujnim algoritmom.

Ovi rezultati predstavljaju primarne lokacije kratkog spoja. Algoritam je uspešno okončan, za sve vrste kratkih spojeva kvar je lociran na zadatoj deonici 1010. Na osnovu rezultata može se zaključiti da je strujna metoda lokacije kvara daleko stabilnija kada su intenziteti struja veći što predstavlja otežavajuću okolnost pri pojavi jednopolnih kratkih spojeva.



Slika 6.1.1. – Rezultati proračuna lokacije kvara strujnom metodom

U tabeli su date sekundarne deonice sa kvarom pri toleranciji merene vrednosti od ±2 [%]. Rezultati potvrđuju da je strujna metoda nestabilna pri malim vrednostima struje kratkog spoja.

Tabela 6.1.2. – Sekundarne deonice sa kvarom pri toleranciji +2 [%]

Tip kratkog spoja	Merena struja kvara [A]	Struja kvara početnog čvora [A]	Struja kvara krajnjeg čvora [A]	Grana
1PKS	274.253	278.617	273.766	1011

Tabela 6.1.3. – Sekundarne deonice sa kvarom pri toleranciji -2 [%]

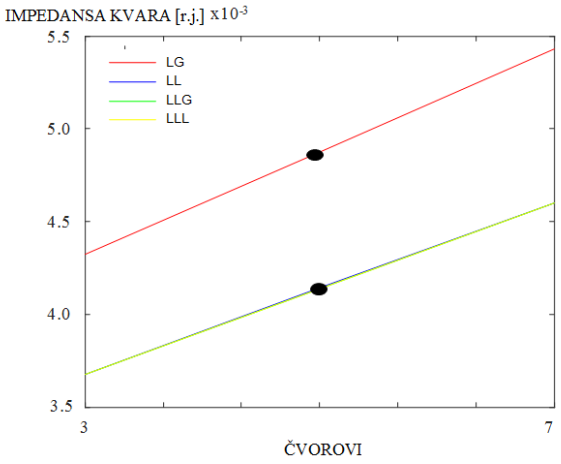
Tip kratkog spoja	Merena struja kvara [A]	Struja kvara početnog čvora [A]	Struja kvara krajnjeg čvora [A]	Grana
1PKS	285.447	291.441	281.102	1009

6.2 Lokacija kvara impedantnom metodom

Tabela 6.2.1 prikazuje rezultate proračuna lokacije kvara impedantnom metodom, na slici 6.2.1 rezultati su predstavljeni grafički.

Tabela 6.2.1 – Rezultati proračuna lokacije kvara impedantnom metodom

Tip kratkog spoja	Merena impedansa kvara [r.j.]	Impedansa kvara čvora 3 [r.j.]	Impedansa kvara čvora 7 [r.j.]	Grana
1PKS	0.0048	0.0043	0.0054	1010
2PKS	0.0041	0.0036	0.0047	1010
2PKSZ	0.0041	0.0036	0.0047	1010
3PKS	0.0041	0.0036	0.0047	1010



Slika 6.2.1. – Rezultati proračuna lokacije kvara impedantnom metodom

Na osnovu rezultata može se zaključiti da je kvar uspešno lociran primenom impedantne metode. Za sva četiri tipa kratkih spojeva kvar je lociran na deonici 1010.

7. ZAKLJUČAK

Procena mesta kvara jedna je od najvažnijih akcija prilikom upravljanja distributivnom mrežom. U ovom radu je obrađeno više različitih metoda za rešenje ovog problema. Na kraju rada izvršeno je poređenje strujne i impedantne metode na primeru jednostavne distributivne mreže. Metode su se pokazale kao pouzdane. Za sve tipove kratkih spojeva kvar je lociran na odgovarajućoj deonici.

8. LITERATURA

[1] Strahil Gušavac, *Osnovni principi projektovanja u mrežama srednjeg i niskog napona,* Fakultet Tehničkih Nauka Novi Sad, 2004.

[2] Duško Bekut, *Relejna zaštita,* Fakultet Tehničkih Nauka Novi Sad, 2009.

[3] B. Brbaklić, *Određivanje optimalnog broja, tipa i lokacije uređaja za automatizaciju elektrodistributivnih mreža,* Doktorski rad, Fakutlet tehničkih nauka Novi Sad, 2018.

[4] F.C.L. Trindade, W. Freitas, J.C.M. Vieira, *Fault Location in Distribution Systems Based on Smart Feeder Meters,* IEEE Transactions on Power Delivery, 2014.

[5] H. Mokhalis, H.Y. Li, A.H, Bakar, *Locating Fault Using Voltage Sag Profile for Underground Distribution System,* IEEE, 2010.

[6] D. Popović, D. Bekut, V. Dabić, *Specijalizovani DMS algoritmi,* DMS GROUP Novi Sad, 2011.

Kratka biografija:

Aleksa Simić rođen je u Novom Sadu 25.01.1994. Osnovne studije na Fakultetu tehničkih nauka završio 2018. godine na smeru Elektroenergetski sistemi. Iste godine upisao master studije, takođe na Fakultetu tehničkih nauka.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.85
DOI: https://doi.org/10.24867/07BE24Basaric

SISTEM ZA PREPOZNAVANJE I PREPORUČIVANJE ODEVNIH PREDMETA SA VIDEO ZAPISA

SYSTEM FOR CLOTHES RECOGNITION AND RECOMMENDATION FROM A VIDEO

Nebojša Basarić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – *Zadatak ovog rada je specifikacija, implementacija i verifikacija sistema za detekciju, prepoznavanje i preporučivanje odevnih predmeta upotrebom složenih arhitektura konvolucionih neuronskih mreža i algoritama u oblasti komjuterske vizije. Za obučavanje modela korišćen je javno dostupan skup podataka objavljen od strane univerziteta u Hong Kongu.*

Ključne reči: *sistemi za preporučivanje proizvoda, analiza slike, kompjuterska vizija, prepoznavanje i detekcija, E-commerce*

Abstract – *This paper describes the specification, implementation, and verification of systems for clothes detection, recognition, and recommendation using the complex convolutional neural network architectures and algorithms in the field of computer vision. DeepFashion dataset, published by the University of Hong Kong, was used to train the model.*

Keywords: *recommender systems, image analysis, computer vision, recognition and detection, E-commerce*

1. UVOD

Online marketing se konceptualno razlikuje od ostalih vidova marketinga time što omogućava direktnu komunikaciju između prodavca i krajnjeg korisnika. Danas je *online* marketing najbrže rastući segment *online* prodaje. Najveća razlika između *online* i tradicionalnog marketinga je obim interakcije između kupca i prodavca. Ovakav vid reklamiranja omogućava kupcima da sami utiču na sadržaj reklama koji će im se prikazivati, kao i prodavcima da svoj proizvod reklamiraju samo potencijalnim kupcima sa određenom sferom interesovanja.

U prethodnoj deceniji, nastala je dramatična izmena u načinu na koji kupci realizuju kupovinu. Iako kupci i dalje fizički posećuju prodavnice, postali su mnogo sigurniji u kupovinu preko interneta. *E-commerce* omogućava ljudima iz malih mesta pristup svim proizvodima u velikim gradovima koji su pre bili nedostupni.

Studija [1] prikazuje najprodavanije tipove proizvoda preko interneta. U ovoj studiji, od 100 kupaca - 33% kupilo je knjige, 23% kupaca koristilo je internet za dobavljanje elektronskih komponenti, 18% kupovalo je odeću, 12% obuću, i 14% ostale tipove proizvoda.

Na osnovu priložene studije može se zaključiti da je kategorija odeće zajedno sa obućom skoro najzastupljenija kategorija prodaje preko interneta. To nas dovodi do motivacije za ovaj rad, odnosno kako dodatno poboljšati interakciju između kupaca i prodavaca u pogledu prodaje odeće. Često nam se dešava da na filmu ili spotu ugledamo osobu koja nosi odeću koja nam se dopada. Prvo što poželimo je da saznamo gde da pronađemo istu takvu ili bar sličnu. Namena sistema predstavljenog u ovom radu jeste da krajnjem korisniku olakša pronalaženje i kupovinu najpogodnijeg odevnog predmeta, sličnog onom koga zapazi na filmu ili u spotu.

U sistemu predstavljenom u ovom radu je kao početni korak vršena detekcija i prepoznavanje tipa odevnog predmeta iz video zapisa. Prepoznavanje se vrši na osnovu jednog frejma videa koristeći konvolucionu neuronsku mrežu (*Convolutional Neural Network*, CNN) *Faster-R CNN* arhitekture. Sistem prepoznaje sledeće klase: majica, haljina, sako, farmerke, sportska majica, venčanica, jakna, suknja i šorc. Nakon prepoznavanja tipa odevnog predmeta proizvodi koji su u ponudi iz te klase se rangiraju po različitim kriterijumima koji definišu sličnost proizvoda: prosečna vrednost boje reprezentovane u dva različita modela i histogram koji označava teksturu. Izračunavanje kriterijuma rangiranja zasnovano je na primeni tehnika kompjuterske vizije.

Za potrebe demonstracije rada ovog sistema implementirana je veb aplikacija koja prikazuje različite video klipove sa interneta i omogućava korisniku da u bilo kom momentu pritiskom na dugme pokrene pretraživanje prodavnica za proizvodom koje se u tom trenutku prikazuje na video klipu.

Implementacija sistema podeljena je na tri modula:

- Modul za Detekciju i prepoznavanje odevnog predmeta sa frejma video zapisa
- Modul za Preporučivanje proizvoda
- Korisnički interfejs

2. METODOLOGIJA

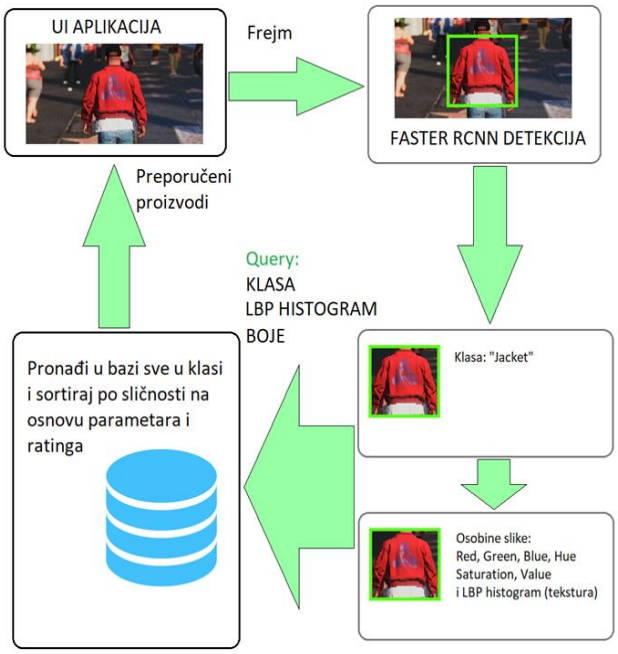
Modeli za detekciju i prepoznavanje odevnih predmeta definisani su, podešavani, trenirani i testirani koristeći *Python* [2] programski jezik i *TensorFlow* [3] bibilioteku koja predstavlja platformu otvorenog koda *state-of-the-art* algoritama mašinskog učenja.

Za implementaciju detaljne analize i segmentacije slike korišten je C# programski jezik na .NET platformi sa *EmguCV* [4] wrapper-om koji predstavlja biblioteku koja omogućuje pozivanje *OpenCV* [5] funkcija iz .NET

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bila dr Jelena Slivka, docent.

okruženja. *OpenCV* (*Open Source Computer Vision Library*) predstavlja biblioteku otvorenog koda za kompjutersku viziju i mašinsko učenje.

Za potrebe prikaza rada sistema implementirana je *web* aplikacija, i to koristeći C# programski jezik na .NET platformi za implementaciju back-end aplikacije, a *JavaScript* i *jQuery* [6] su korišćeni za implementaciju klijentske aplikacije.



Slika 2.1. Arhitektura rešenja

Na slici 2.1. je predstavljena skica arhitekture rešenja predloženog u ovom radu gde se mogu prepoznati sva tri modula koja čine sistem: Modul za detekciju i prepoznavanje odevnih predmeta, modul za preporučivanje *n* najsličnijih proizvoda i modul koji predstavlja korisničku aplikaciju koja prikazuje rad sistema.

2.1. Modul za prepoznavanje i detekciju odevnih predmeta

Zadatak modula za prepoznavanje i detekciju odevnih predmeta jeste da prihvati ulaznu sliku sa korisničkog interfejsa i na njoj pronađe odevne predmete, odnosno *bounding box* objekte (*bounding box* objekat sadrži koordinate gornje leve tačke i donje desne tačke pravougaonika) u okviru kojih se nalazi odevni predmet koji pripada jednoj od devet razmatranih klasa.

Za realizaciju ovog modula korištena je *Faster* R-CNN konvoluciona neuronska mreža, sa arhitekturom *Faster* RCNN *Inception* v2 [7]. Korišćena arhitektura je integrisana u *Tensorflow* biblioteku na detection model zoo repozitorijumu koji predstavlja kolekciju prethodno treniranih modela nad COCO skupom podataka. Ova arhitektura je odabrana na osnovu rada [8] koji se bavio sličnom temom detekcije odevnih predmeta. Ovaj rad je imao veoma uspešne rezultate i, po ugledu na njega, inicijalna arhitektura neuronske mreže koja je isprobana je *Faster RCNN Inception Resnet v2*.

Ova arhitektura je davala precizne rezultate na sistemu prezentovanom u ovom radu, međutim, zbog veoma male brzine bila je skoro neupotrebljiva. Zbog toga, odlučeno je da se koristi arhitektura *Faster* RCNN *Inception* v2 koja predstavlja model sa najboljim odnosom brzine i preciznosti.

Arhitekturu izabranog modela čine dva konvoluciona sloja, i dva sloja sažimanja koja primenjuju max pooling funkciju. Za normalizaciju se koristi ReLu funkcija [9]. Model je prethodno treniran na COCO skupu podataka koji sadrži opšte slike, i dotreniran je nad skupom podataka koji sadrži okvirno 400-500 slika odeće po klasi, što je ukupno oko 4000 slika (veći broj slika bi bio poželjan, ali je ovaj broj izabran kao kompromis vremena i kvaliteta treniranja – slike su ručno uređivane, i vremenski je zahtevno prolaziti kroz ceo *DeepFashion* [10] skup). Dotreniravanje je vršeno u 18 epoha (veći broj epoha bi dao bolje rezultate, međutim, zbog vremenskog ograničenja treniranje je zaustavljeno, i pošto su rezultati bili zadovoljavajući, nije bilo neophodno nastaviti treniranje).

Prilikom dotreniravanja ostavljeni su zamrznuti konvolucioni slojevi koji sadrže već podešene težine na osnovu slika iz COCO skupa, a odmrznuti su poslednji potpuno povezani slojevi namenjeni za klasifikaciju. Time je postignuto da konvoluciona mreža koja je već naučena da prepoznaje šablone sa slike, zna da klasifikuje sliku u devet klasa za potrebe ovog sistema.

Izlaz iz ovog modula su koordinate odevnog predmeta sa slike, odnosno, koordinate pravougaonika u kom se nalazi, kao i klasa kojoj predmet pripada.

2.2. Modul za preporučivanje

Ulaz u modul za preporučivanje predstavlja izlaz iz prethodnog modula, odnosno, deo slike koji predstavlja odevni predmet. U ovom delu sistema se bavimo tehnikama kompjuterske vizije, i oslanjamo se na *OpenCV* biblioteku.

Kako bismo izvukli relevantne parametre sa odevnog predmeta, moramo eliminisati pozadinu. Za to koristimo *Canny Edge detector* [11] i tražimo najveću konturu na slici. S obzirom na to da smo izvršili selekciju regiona od interesa u prethodnom koraku, pretpostavka je da je najveća kontura sam odevni predmet. Nakon što smo izolovali odevni predmet sa slike možemo da pristupimo procesu izdvajanja sledećih parametara:

- Prosečna vrednost *Red, Green i Blue* komponente izdvojenog regiona
- Prosečna vrednost Hue, Saturation i Value komponente izdvojenog regiona
- Koristeći lokalni binarni obrazac, dobijamo histogram koji predstavlja teksturu odevnog predmeta.

Do preporuke prvih *n* proizvoda dolazi se na jednostavan način. Vrši se upit u bazu gde se zahtevaju proizvodi iz iste klase kao detektovani predmet, sortirani po sličnosti prosleđenih parametara (odnosno, gde je apsolutna vrednost razlike parametara najmanja). Mora se uzeti u obzir da nemaju svi parametri isti uticaj, pa tako *HSV* boja mnogo više utiče na računanje sličnosti nego *RGB*. To je regulisano tako što se apsolutna vrednost razlike *HSV* parametara množi određenom vrednošću kako bi imala veću težinu u procesu preporučivanja. Do konačne

vrednosti tog fakotra došlo se empirijski posmatranjem ponašanja sistema.

2.3. Grafički korisnički interfejs

Korisnički interfejs aplikacije je implementiran kao *web* aplikacija koja omogućava korisnicima sledeće osnovne funkcionalnosti:

- Pregled video zapisa, pretragu i navigaciju različitih video zapisa
- U bilo kom momentu trajanja video zapisa, pritiskom na dugme korisnici mogu da pretraže bazu proizvoda u potrazi za odevnim predmetom koji se u tom trenutku prikazuje na videu
- Pregled različitih proizvoda i ocenjivanje istih.

2.4. Skup podataka

Za obučavanje i testiranje modela konvolucione neuronske mreže čiji zadatak je detekcija i klasifikacija odevnih predmeta u devet kategorija, korišten je *DeepFashion dataset* [8] koji je sačinjen od preko 800.000 kvalitetnih slika (različitih dimenzija) na kojima su odevni predmeti dobro vidljivi jer se radi o slikama iz *online* prodavnica, gde su poze pogodne tako da odeća dolazi do izražaja.
Zbog hardverskih ograničenja, za potrebe ovog projekta nije korišten ceo skup podataka, već samo oko 4 000 slika, raspoređenih u devet klasa.
Neke slike sadrže opis teksture koji je omogućio evaluaciju podsistema za preporučivanje (npr., *Striped_Maxi_Dress, Floral_Side-Slit_Maxi_Dress, Leopard_Print_Maxi_Dress, Geo_Striped_Maxi_Dress*).
Problem koji postoji kod ovog skupa podataka jeste da ima previše klasa koje sadrže veoma slične podatke. Iz tog razloga je rađeno ručno prolaženje kroz klase skupa podataka i više klasa je spajano u jednu (na primer, postoji nekoliko klasa farmerki, ali su razlike neprimetne čak i čoveku kao posmatraču, tako da su ove klase spojene u jedinstvenu klasu - farmerke).
Spajanjem sličnih klasa i izbacivanjem klasa koje imaju previše šuma (postoje klase koje imaju toliko različitih slika da im čak ni čovek ne može naći sličnost, npr. klasa Pepulum_top) došlo se do devet klasa razmatranih u ovom radu.

3. EVALUACIJA

Dva osnovna modula koja čine ovaj sistem (modul za detekciju i prepoznavanje i modlul za preporučivanje proizvoda) su evaluirani kao potpuno nezavisne komponente nad različitim delovima *DeepFashion* skupa podataka.
Za evaluaciju modula za prepoznavanje i detekciju korišćene su slike različitih klasa, a za modul za preporučivanje korišćene su slike predmeta iste klase, ali različite teksture i boje.

3.1. Evaluacija modula za detekciju i prepoznavanje

Za potrebe evaluacije modela *Faster* RCNN konvolucione neuronske mreže, skup podataka je podeljen na trening skup (85%) i test skup (15%). Kao mera evaluacije korišćena je F-mera (*F1-score* metoda) i postignuti su zadovoljavajući rezultati, s obzirom na kvalitet skupa podataka i hardverska ograničenja.

Kategorija	F-mera
Majica	82%
Haljina	80%
Sako	89%
Farmerke	96%
Sportska majica	92%
Venčanica	86%
Jakna	92%
Suknja	70%
Šorc	87%

Tabela 3.1. Rezultati evaluacije modela za detekciju i prepoznavanje

Kada posmatramo tabelu 3.1, iako nema velikih odstupanja u zavisnosti od klase, možemo izdvojiti sledeća zapažanja:

- Najveća preciznost postignuta je kod farmerki - razlog za to je najverovatnije što su farmerke jedini odevni predmet koji predstavlja dugački donji deo odeće. Kada bismo uneli klasu pantalone pored farmerki, verovatno bismo značajno narušili preciznost.
- Najlošije rangirana klasa je suknja sa 70% tačnosti. Posmatranjem rezultata testa, zaključeno je da se suknja veoma često pogrešno klasifikuje kao haljina ili venčanica.
- Primećeno je često pogrešno prepoznavanje šorca na određenim slikama gde se ne vidi ceo donji deo tela.

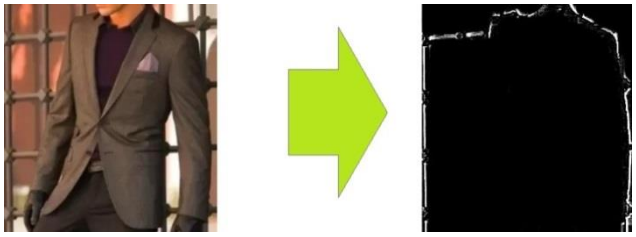
3.2. Evaluacija modula za preporučivanje

Celokupan modul za preporučivanje koji bi uključivao ocene pojedinačnih korisnika, te time pružio mogućnost razvoja personalizovanog sistema za preporuku za sada nije moguće precizno evaluirati. Međutim, evaluiran je deo sistema koji je najkompleksniji u ovom modulu, a to je *LBP* operator zadužen za ekstrakciju teksture sa slike. Ostala obeležja koja označavaju boju nisu uzeta u obzir zato što ne postoje odgovarajuće anotacije u skupu podataka koje bi omogućile evaluaciju preporuke koja uzima u obzir i za obeležja.

Kako bi se uspešno izvršila evaluacija LBP operatora, problem preporučivanja je predstavljen kao problem klasifikacije. *DeepFashion dataset* sadrži atribute koji opisuju teksturu, tako da je to omogućilo evaluaciju LBP operatora.

U bazi proizvoda su sačuvana četiri tipa haljina različitih tekstura: leopard tekstura, tačkasta tekstura, tekstura na horizontalne linije i tekstura na vertikalne linije. Testiranje je vršeno tako što se za prosleđeni tip teksture od sistema za preporuku očekuje da preporuči haljinu sa istim dezenom. Ovo je moguće izvršiti i na ostalim klasama, ali je odlučeno da se koristi klasa haljine jer ima najviše različitih tipova tekstura. U procesu evaluacije ovog podsistema uspešnom preporukom se smatra slučaj kada istu oznaku teksture ima slika na ulazu kao i slika u bazi proizvoda koja je preporučena kao najpogodnija. Na tačnost evaluacije ovog podsistema ne utiče prethodni korak, odnosno klasifikacija. Svi proizvodi koji ulaze u evaluaciju priparadju istoj klasi. Kao mera evaluacije korišćena je F-mera i postignut je rezultat od 76%.

Analizom grešaka modela utvrđeno je da je u ovom najslabija karika pronalaženje najveće konture. Problem se najčešće javlja kada je iza detektovanog objekta previše šarena pozadina, te se deo pozadine uzme kao površina za ekstrakciju atributa boje i teksture (slika 3.1).
Može se desiti da je kontrast između odevnog predmeta i pozadine previše mali tako da se ivice neuspešno detektuju i u tom slučaju se uzima ceo *bounding box* kao površina od interesa što vodi ka nepreciznom računanju prosečne boje i histograma teksture.



Slika 3.1. Pogrešno detektovana najveća kontura na slici

4. ZAKLJUČAK

U ovom radu je predstavljen sistem za prepoznavanje i preporučivanje odevnih predmeta sa video zapisa. Motivacija za takav sistem potekla je iz sfere *online* prodaje i marketinga. Osnovna ideja je bila da se napravi sistem koji pomaže potencijalnim kupcima odeće preko interneta da pronađu kombinaciju koju vide na filmu, a samim tim i prodavcima da lakše prodaju svoje proizvode. Sistem je sačinjen iz tri modula: modul za detekciju i prepoznavanje (ulaz u modul predstavlja slika dobijena sa korisničkog interfejsa, a izlaz je *bounding box* koji govori o tome gde se na slici nalazi detektovani proizvod), modul za preporučivanje proizvoda (ulaz predstavlja *bounding box* iz prethodnog modula, a izlaz najsličnijih *n* proizvoda), i grafički korisnički interfejs kao odvojena aplikacija.

Prvi modul se oslanja na *Faster* RCNN konvolucionu neuronsku mrežu kreiranu sa *Faster* RCNN *inception v2* arhitekturom. Za evaluaciju prvog modula korištena je F-mera koja je za ovaj modul iznosila 86% tačnosti. Za obučavanje i testiranje veštačke neuronske mreže korišten je *DeepFashion* skup podataka. Osnovni problem korišćenog skupa podataka je bio preveliki broj kategorija, tako da je skup morao biti ručno prerađen za potrebe ovog sistema.

Drugi modul se oslanja na različite metode kompjuterske vizije (*Canny detektor ivica, LBP* operator, itd.) i zasniva se na ekstrakciji boje i teksture. Ovaj podsistem nije pogodan za dubinsku evaluaciju usled nedostataka odgovarajućih anotacija skupa podataka, ali je svakako evaluiran LBP operator koristeći F-meru i postignuta F-mera iznosi 76%.

Prilikom evaluacije, uočeni su i neki nedostaci sistema: Zbog hardverskih ograničenja nije iskorišten ceo obučavajući skup, i rađen je mali broj epoha pri obučavanju. Duže obučavanje bi verovatno unapredilo performanse sistema.

Drugi problem sistema je što vrši detekciju samo jednog proizvoda sa video zapisa i ukoliko bi više odevnih predmeta bilo prikazano u isto vreme, sistem bi sam odabrao predmet koji će se preporučiti, odnosno, korisnik ne može da utiče na to. *R-CNN* mreža svakako pronalazi više predmeta, međutim samo jedan se uzima u obzir (onaj koji je prepoznat sa najvećom sigurnošću) kako bi se u aplikaciji prikazali različiti proizvodi iste vrste i korisnik imao mogućnost upoređivanja.
Problem je neodstatak opcije da korisnik selektuje konkretan odevni predmet.

5. LITERATURA

[1] Dr.R.Shanthi1 Dr. Desti Kannaiah, Vol.13, 2015, Consumers' Perception on Online Shopping, Assistant Professor, Department of Commerce, University of Madras, Chennai -600005, Tamil Nadu,India

[2] Python.org. (2019). Welcome to Python.org. Available at: https://www.python.org/ [15 Oct. 2019].

[3] TensorFlow. (2019). TensorFlow. Available at: https://www.tensorflow.org/ [15 Oct. 2019].

[4] Emgu.com. (2019). Emgu CV: OpenCV in .NET (C#, VB, C++ and more). Available at: http://www.emgu.com/wiki/index.php/Main_Page [15 Oct. 2019].

[5] Opencv.org. (2019). OpenCV. Available at: https://opencv.org/ [15 Oct. 2019].

[6] js.foundation, J. (2019). jQuery. Jquery.com. Available at: https://jquery.com/ [15 Oct. 2019].

[7] Tensorflow detection model zoo https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md

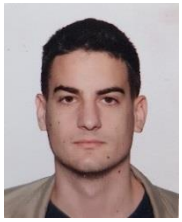
[8] Clothing Detection for Fashion Recommendation https://blog.valohai.com/clothes-detection-for-fashion-recommendation

[9] Yun Ren,Changren and Shunping Xiao, Object Detection Based on Fast/Faster RCNN Employing Fully, Convolutional Architectures, Hindawi Mathematical Problems in Engineering Volume 2018

[10] Large-scale Fashion (DeepFashion) Database, Multimedia Laboratory, The Chinese University of Hong Kong http://mmlab.ie.cuhk.edu.hk/projects/DeepFashion.html

[11] Bill Green, Canny Edge Detection Tutorial (2002), Drexel Autonomous Systems Lab., 2002. http://masters.donntu.org/2010/fknt/chudovskaja/library/article5.htm

Kratka biografija:



Nebojša Basarić rođen je 18.05.1994. godine u Kninu, Republici Hrvatskoj. 2009. godine upisuje Srednju tehničku školu u Somboru, smer elektrotehničar računara. Srednju školu završava 2013. godine kada i upisuje osnovne studije na Fakultetu tehničkih nauka u Novom Sadu, smer računarstvo i automatika. Osnovne studije završava u roku 2017. godine i upisuje master studije na istom fakultetu. Položio je sve ispite predviđene planom i programom. kontakt: nebojsabasaric94@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.85
DOI: https://doi.org/10.24867/07BE25Peric

DETEKCIJA SARKAZMA U KOMENTARIMA SA *REDDIT* STRANICE

SARCASM DETECTION IN *REDDIT* COMMENTS

Sara Perić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Analiza sentimenta je veoma zastupljena u istraživanjima danas. Deo problema određivanja sentimenta predstavlja detekcija sarkazma, jer on često može da navede model da zaključuje suprotno od tačnog. Ovaj fenomen se javlja zbog same prirode sarkazma: upotreba pozitivnih reči u cilju izražavanja negativnih osećanja. Tema ovog rada je detekcija sarkazma u komentarima, gde se on češće javlja u odnosu na druge tekstualne sadržaje. U ovom radu prikazani su različiti pristupi u rešavanju datog problema. Predloženi su različiti klasifikacioni modeli – metod slučajnih šuma (engl. Random Forest), metod potpornih vektora (engl. Support Vector Machines - SVM), logistička regresija, kao i različite arhitekture neuronskih mreža – Yoon Kim model, konvolucioni model, rekurentni model i konvolucioni-rekurentni. Uporedo sa srodnim istraživanjima i ovim radom je pokazano da je detekcija sarkazma moguća i da se daljim unapređenjem modela tačnost može povećati i time doprineti značajnom poboljšanju analize sentimenta.*

Ključne reči: *Sarkazam, Konvolucione i Rekurentne neuronske mreže, Metod potpornih vektora, Logistička regresija, Metod slučajnih šuma*

Abstract – *Sentiment analysis is widespread in research today, and sarcasm represents one of its problems, which can often lead the model to conclude the opposite of the correct sentiment. This phenomenon occurs because of the very nature of sarcasm: the use of positive words to express negative feelings. Therefore, the topic of sarcasm detection was chosen particularly in comments, where it occurs more often than in usual textual content. This paper presents different approaches to solving a given problem. Different classification models are proposed - Random Forest, Support Vector Machines (SVM), Logistic Regression, as well as various neural network architectures - Yoon Kim model, convolution model, recurrent model, and convolution-recurrent model. Along with related research, this work has shown that sarcasm detection is possible and that by further refining of the model, accuracy can be increased and thus contribute to a significant improvement in sentiment analysis.*

Keywords: *Sarcasm, Convolutinal and Recurrent neural networks, Support Vector Machines, Logistic Regression, Random Forest*

1. UVOD

Sarkazam predstavlja stilsku figuru koja se koristi za izražavanje negativnih osećanja korišćenjem pozitivnih reči. Tumači se kao ruganje s ciljem da se ismeje onaj kome je upućena sarkastična poruka. Tokom dijaloga, ljudi se često koriste tehnikama naglašavanja reči, kao i različitim oblicima gestikulacije s ciljem naglašavanja ironije. S obzirom na to da se u tekstu ne nalazi takav vid „olakšice“ za prepoznavanje sarkazma, utoliko je teže detektovati ga, kako za ljude tako i za istrenirane modele mašinskog učenja. Performanse sentiment analize, koja predstavlja deo mnogih modernih istraživanja, mogu biti narušene zbog problema pogrešnog zaključivanja sentimenta usled prisustva sarkazma [1].

Istraživanjem rešenja koja se bave problemom detekcije sarkazma, došlo se do zaključka da je komentare pre svega potrebno pretprocesirati, a da se potom nad njima treniraju različiti modeli mašinskog učenja. Naučni radovi koji se bave sličnom temom su vršili detekciju sarkazma u postovima sa *Twitter*[1] društvene mreže [6, 7, 8, 10]. Za razliku od njih, ovaj rad koristi drugačiji skup podataka, sa drugog veb portala (*Reddit*[2]).

Dati skup podataka se sastoji od komentara varijabilne dužine (od po koje reči do nekoliko rečenica), tekst je manje formalan, te i ispravnost gramatike varira od komentara do komentara. Dok su tvitovi celina za sebe, komentari sa *Reddit* stranice mogu biti uslovljeni samim sadržajem koji se komentariše, tekstom roditeljskog komentara te i viših komentara u hijerarhiji vezanih za originalni post.

Ono što je u ovom radu slično prethodnim rešenjima jesu metode koje su bile primenjene za rešavanje datog problema, a to su: metod slučajne šume (*Random Forest*) [2], metod potpornih vektora (*Support Vector Machines*, SVM) [3], logistička regresija (*Logistic Regression*) [4], kao i *Yoon Kim* model [5], konvolucioni model i konvoluciono-rekurentni model.

U narednom poglavlju su prikazani srodni radovi koji se bave problemom detekcije sarkazma u tekstu, modeli koji one koriste i rezultati koji su postignuti.

U poglavlju 3 opisan je skup podataka, komentara sa *Reddit* stranice koji je korišćen u radu, te primenjene metodologije.

U poglavlju 4 prikazani su postignuti rezultati. Poglavlje 5 sadrži sumarizaciju rada.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bila dr Jelena Slivka, docent.

[1] https://www.twitter.com
[2] https://www.reddit.com

2. SRODNA ISTRAŽIVANJA

Zadatak rada [6] jeste prepoznavanje sarkazma u postovima na *Twitter*-u, a rešenje je bazirano na prethodnom istraživanju [7]. Skup podataka je nastao tokom nekoliko meseci skupljanja tvitova 2014. godine. Sadrži 25.278 sarkastičnih i 117.842 nesarkastičnih postova. Sakupljeni postovi su nasumice podeljeni na dva skupa, 70% podataka za trening i 30% za test skup. Osnovni model korišten u radu [6] je SVM, na način na koji je implementiran u *LinearSVC*[3] funkciji iz biblioteke *scikitlearn*[4] u *Python*-u. Drugi pristup isproban u radu [6] je primena Naivnog Bajesovog klasifikatora[5], dok je treći bio takođe SVM, ali samo sa jednom klasom koju su činili nesarkastični postovi. Pre obuke modela mašinskog učenja je izvršeno pretprocesiranje podataka, pri čemu su izbačeni haštagovi, karakteri koji nisu ASCII i http linkovi. Za evaluaciju rešenja su korišteni F1 mera i tačnost (eng. *accuracy*).

Tema rada [8] je prepoznavanje sarkazma u tekstualnom sadržaju, na osnovu grupe ključnih reči izdvojenih iz *Twitter* postova. Skup podataka je generisan tako što su izdvojeni postovi koji imaju u sebi *#sarcasm* ili *#sarcastic* i oni su dati ljudima na prevođenje, s ciljem pronalaska adekvatne reči u bukvalnom značenju (za razliku od sarkazma prisutnog u inicijalnoj reči). Za ovo je korišćena *Amazon Mechanical Turk* platforma[6]. Na ovaj način je za svaki sarkastičan tvit dobijeno više nesarkastičnih prevoda, nakon čega je tehnikama nenadgledanog učenja detektovana ključna reč iz sarkastičnog tvita, traženjem reči sa suprotnim značenjem.

Nakon što je kreiran skup ključnih reči, skup sarkastičnih izjava je dobijen tako što su iz skupa sarkastičnih tvitova (*#sarcasam* ili *#sarcastic*) izdvojeni oni u kojima se ključne reči pojavljuju, a skup nesarkastičnih izjava izdvajanjem tvitova bez pomenutih tagova koji sadrže ključne reči. Izdvojen je i treći skup, koji predstavlja skup nesarkastičnih izjava sa sentimentom, tako što su traženi tvitovi sa tagovima poput *#sad*, *#happy* koji sadrže neku ključnu reč. Za 70 ključnih reči ukupno je prikupljeno 2.542.249 tvitova. Ukupno 80% je korišteno za trening, 10% za test i 10% za razvoj. Primenjena su dva pristupa, distribucioni i klasifikacioni. Distribucioni semantički model koristi kontekstne vektore za reprezentaciju podataka i kosinusnu sličnost kao meru udaljenosti.

Za klasifikaciju je korišten SVM sa modifikovanim kernelom i različitim *word embedding* modelima [9]. Tvitovi su pretprocesirani tako da su sve reči konvertovane u mala slova, izbačeni su hashtagovi, svi brojevi su konvertovani u generički token "22". Ono što se razlikuje od rada [6], je *word2vec* koji je usvojen i iskoristićen i u modelu prikazanom u ovom radu, kao deo pripreme podataka, što je kasnije objašnjeno u četvrtom poglavlju.

Zadatak rada [10] jeste prepoznavanje sarkazma u tekstualnom sadržaju tvitova. Metodologija za rešavanje problema se sastojala u korišćenju dubokih konvolucionih mreža sa ciljem uočavanja konteksta samog tekstualnog sadržaja. Upotrebljeno je više nezavisno istreniranih modela konvolucionih neuronskih mreža namenjenih za predikciju sentimenta (skup podataka je sadržao rečenice od kojih su 5895 pozitivnog sentimenta, 3131 negativnog i 471 neutralnog), emocije teksta (sa skupom od 5205 rečenica) i ličnosti (sa skupom podataka od 2400 eseja labeliranih tipovima ličnosti). Krajnje rešenje koristi dva modela: „čistu" konvolucionu neuronsku mrežu i obeležja izdvojena iz konvolucione mreže koja su potom prosleđena kao ulaz u SVM model. Prilikom pripreme podataka, za reprezentaciju reči korišten je *Google*-ov *word2vec* [9]. Ono što se može zaključiti iz ovog rada jeste da se za rešavanje problema detekcija sarkazma mogu koristiti konvolucione neuronske mreže, te su po uzoru na njega primenjene i u ovom radu. Za računanje tačnosti korištena je F1 mera, koja je iznosila 93,30%.

Po uzoru na rad [6], u ovom radu je primenjen SVM, sa različitim parametrima, o čemu je reč u poglavlju 3. Naivni Bajes nije davao veću tačnost od SVM-a, te nije korišćen u ovom radu. Obrada podataka iz rada [6] je uzeta kao osnovna ideja, koja je potom nadograđena. Ideja za uptrebu *word2vec* modela proistekla je iz rada [8] i [10]. Rad [10] je pokazao da se za rešavanje problema detekcije sarkazma mogu koristiti konvolucione neuronske mreže, te su po tom uzoru primenjene i u ovom radu.

3. METOD

U narednim poglavljima izloženi su skup podataka, arhitektura i trening modela.

3.1. Skup podataka

Korišćeni skup podataka je javno dostupan resurs sa *Kaggle*[7] sajta, koji se odnosi na komentare na *Reddit*-u[8]. Čini ga potpuno balansirani odnos broja sarkastičnih komentara prema nesarkastičnim, odnosno, 505.413 komentara svakog tipa. Ciljni atribut predstavlja labela kao indikator prisutnosti sarkazma u komentaru, dok ostale atribute predstavljaju: roditeljski komentar, tekst komentara, *subreddit* (označava kom podforumu pripada komentar), *ups* i *downs* (numerički atributi koji pokazuju koliko osoba je dalo pozitivnu, odnosno, negativnu ocenu posmatranom komentaru), s*core* (numerički atribut koji predstavlja ukupnu ocenu komentara), autor komentara, vreme i datum kreiranja komentara.

3.2. Obrada podataka

Inicijalna faza razvoja bazirana je na pretprocesiranju skupa podataka i obradi teksta, te obuhvata:

1. Izbacivanje nepoželjnih karaktera (linkova, tagova, ne alfa-numeričkih karaktera, zamena pojave karaktera @ sa "at"),
2. Prebacivanje teksta u mala slova
3. Tokenzaciju - pretvaranje komentara i roditeljskog komentara u dva vektora stringova; odvajanje skraćenica, većine znakova interpunkcije, zareza i pojedinačnih navodnika, kada ih prati razmak, kao i tačaka koje se nalaze na kraju
4. Izbacivanje stop reči - reči koje se često javljaju u tekstu, a ne doprinose značenju teksta (zamenice, predlozi i prilozi kao što su, za dati skup podataka na

[3] https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html
[4] https://scikit-learn.org/stable/index.html
[5] https://scikit-learn.org/stabsle/modules/naive_bayes.html
[6] https://www.mturk.com/
[7] https://www.kaggle.com/
[8] https://www.kaggle.com/danofer/sarcasm

engleskom jeziku „*the*", „*a*", „*an*", „*in*", „*between*", „*yourself*", „*but*", „*again*", itd.); stop reči su preuzete iz *nltk*[9] biblioteke, iz kategorije reči za engleski jezik.

3.3. Metodologija

Metodologija ovog rešenja može se podeliti u dve celine: obučavanje i upotreba različitih vektorskih reprezentacija teksta (u ovom slučaju: komentara i roditeljskog komentara) i korišćenje odgovarajuće klasifikacione metode u cilju pronalaženja one koja za dati problem daje najtačnije rezultate. Isprobane vektorske reprezentacije su *word2vec* [9] i *GloVe* [11].

Za potrebe klasifikacije je izvučeno 100.000 nasumičnih primeraka iz dobijenog modela, kako bi računska zahtevnost procesa klasifikacije bila manje kompleksna. Trening i test skupovi su dobijeni deljenjem originalnog skupa podataka, u razmeri 7:3 pri čemu je očuvana distribucija labela.

Word2vec model je istreniran nad celokupnim skupom podataka (komentari i roditeljski komentari). Postavljeni parametri istreniranog modela su sledeći:

- Dimenzionalnost vektora: 50
- Minimalni broj pojavljivanja reči u skupu: 1
- Maksimalna distanca između trenutne i prediktovane reči u rečenici: 10
- Vrednost praga iznad kog reči sa većom frekvancijom se nasumično smanjuju: $1e^{-3}$.

Ostali parametri imaju predefinisane vrednosti. Razlog ovakvih podešavanja se ogleda u optimalnim rezultatima do kojih se došlo empirijskom analizom. Model je prethodno treniran i sa drugačijim parametrima, od kojih je bitno istaknuti dimenzionalnost vektora, čija vrednost je prvobitno bila postavljena na 300, 150, 100. No, kako na taj način istrenirani model nije doprinosio tačnosti rešenja, odabrana je vrednost 50, što je uzrokovalo ubrzanjem treniranja i predikcije rešenja. Pored ručno treniranog modela, isproban je i pretrenirani model *Google News*, dimenzionalnosti 300 [12].

GloVe nudi nekoliko pretreniranih modela od kojih su isprobana dva trenirana nad *Twitter* sadržajima, dimenzionalnosti vektora 50 i 100 [11].

Sledi pregled klasifikacionih metoda.

3.3.1 Yoon Kim model – NN model 1

U ovom projektu korištena je modifikovana verzija *Yoon Kim* modela. Implementiran je korišćenjem *Keras* programske biblioteke[10]. Sastoji se od:

1. Ulaznog sloja (*Embedding*)
2. Konvolucionog sloja (veličina filtera: 128, veličina kernela: 3, aktivaciona funkcija: *Relu*)
3. *MaxPooling* sloja
4. Konvolucionog sloja (veličina filtera: 128, veličina kernela: 4, aktivaciona funkcija: *Relu*)
5. *MaxPooling* sloja
6. *Droput* sloja
7. Potpuno povezanog sloja (veličina: 128)
8. *Dropout* sloja
9. Potpuno povezanog sloja (aktivaciona funkcija: *softmax*, optiomizaciona funkcija: Adam)

Korišćeni atributi: labela, komentar, roditeljski komentar. Kako upotreba ostalih atributa nije uticala na povećanje tačnosti, pri daljem treniranju modela nisu korišteni. Isto važi i za ostale NN (*Neural Network*) modele.

3.3.2 Konvolucioni model – NN model 2

Kreiran po uzoru na metod koji je korišćen u radu [10]. Sastoji se od:

1. Ulaznog sloja (*Embedding*)
2. Konvolucionog sloja (veličina filtera: 128, veličina kernela: 4, aktivaciona funkcija: *Relu*)
3. *MaxPooling* sloja
4. Konvolucionog sloja (veličina filtera: 128, veličina kernela: 3, aktivaciona funkcija: *Relu*)
5. *MaxPooling* sloja
6. Potpuno povezanog sloja (veličina: 128)
7. Potpuno povezanog sloja (veličina: 2, aktivaciona funkcija: *softmax*, optimizaciona funkcija: *Adadelta*)

3.3.3 Konvoluciono-rekurentni model – NN model 3

Sastoji se od:

1. Ulaznog sloja (*Embedding*)
2. Konvolucionog sloja (veličina filtera: 64, veličina kernela: 5, aktivaciona funkcija: *Relu*)
3. *MaxPooling* sloja
4. LSTM (*Long shot-term memory*) sloja (veličina: 100)
5. Potpuno povezanog sloja (aktivaciona funkcija: *softmax*, optimizaciona funkcija: *Adagrad*

3.3.4 Rekurentni model – NN model 4

Isporban metod biderekcionih LSTM ćelija:

1. Ulazni sloj (*Embedding*)
2. Bidirekcioni LSTM sloja (veličine:100)
3. Potpuno povezanog sloja (aktivaciona funkcija: *softmax*, optimizaciona funkcija: rmsprop).

3.3.5 Ostale klasifikacione metode

Pored navedenih modela neuronskih mreža, isprobane su i druge tehnike mašinskog učenja po ugledu na [6, 8, 10]. Korišteni atributi: labela, komentar, roditeljski komentar, *score*, *ups*, *downs*. Do vrednosti navedenih parametara u zagradama došlo se empirijski, korišćenjem validacionog skupa, dok ostali parametri zadržavaju podrazumevane vrednosti.

1. *Random Forest* klasifikator (*n_estimators*: 1000, *min_samples_split*: 16)
2. SVM klasifikator (LinearSvc, c: 0.01)
3. Logistička regresija (*multiclass*: "ovr", *solver*: "liblinear").

4. REZULTATI I DISKUSIJA

Treniranje modela neuronskih mreža je zaustavljeno *Keras* mehanizmom za rano zaustavljanje (engl. *EarlyStopping*) posmatranjem funkcije gubitka (engl. *loss function*), pri promeni manjoj od 0,001 i faktorom strpljenja 6.

Najbolje se pokazao NN model 1, korišćenjem *word2vec* vektorske reprezentacije veličine 50, treniranog nad celokupnim skupom podataka. Tokom 12 epoha se trenirala

[9] https://www.nltk.org/
[10] https://keras.io/

mreža dostignuvši tačnost od 66% nad test skupom podataka, a 72% nad trening skupom podataka.

Tokom 9 epoha trenirala se mreža modela NN 2 dostignuvši tačnost od 63% nad test skupom podataka, a 80% nad trening skupom podataka, korišćenjem *word2vec* vektorske reprezentacije veličine 50, trenirane nad celokupnim skupom podataka.

Najvišu tačnost model NN 3 je dostigao korišćenjem *GloVe* vektorske reprezentacije, veličine 100, prethodno istreniran nad *Twitter* sadržajima. Mreža je trenirana tokom 9 epoha, dostignuvši tačnost od 63% nad test skupom podataka, a 76% nad trening skupom podataka.

NN Model 4 se najbolje pokazao korišćenjem *GloVe* vektorske reprezentacije veličine 100, prethodno istreniran nad *Twitter* sadržajima. Mreža je trenirana tokom 15 epoha, dostignuvši tačnost od 64% nad test skupom podataka, a 78% nad trening skupom podataka.

Random Forest se najbolje pokazao ne koristivši se atributom roditeljskog komentara sa tačnošću 65,31%. SVM daje gore rezulate: 62,02%, dok mu je vrlo bliska Logistička regresija sa dostignutih 62,03%.

Tačnost koja je postignuta u ovom radu je u opsegu 60% do 80%, zavisno od modela koji je upotrebljen, nasuport najvišoj tačnosti od 97% u srodnim radovima.

Mimoilaženja u rezultatima ogledaju se delom u veličini skupa podataka, a delom i u samom kvalitetu podataka. Komentari su, poput postova, nestruktuirani tip podataka, ali ono što dodatno otežava detekciju sarkazma u njima jeste što se, za dati skup podataka, sarkazam može odnositi na sadržaj samog komentara, na roditeljski komentar ili čak komentar koji je izvan domena postojećeg skupa podataka. Istrenirani modeli korišteni u radu [10] dosta su kompleksniji od modela do kojih se došlo u ovom radu. Pored toga, sam skup podataka nad kojim je obučena mreža dosta se razlikuje, s obzirom da je pored tvitova korišten veliki broj eseja, dubokog teksta, što samom modelu „olakšava“ zaključivanje konteksta.

Kao jednu od mogućnosti unapređenja, potrebno je naglasiti važnost pronalaženja novih specifičnih obeležja sarkastičnog teksta, pomoću kojih bi detektovanje postalo preciznije. Potrebno je i naći veće i pouzdanije skupove podataka, koji zasigurno sadrže sarkastične tekstove, za razliku od oslanjanja na *#sarcasm*, *#sarcastic* i slična labeliranja od strane samih korisnika društvenih mreža.

5. ZAKLJUČAK

U ovom radu predstavljeno je nekoliko metoda mašinskog učenja koje bi se mogle primeniti za rešavanje problema detekcije sarkazma kao i kako se svaki od njih ponaša na zadatom skupu komentara sa *Reddit* veb stranice. Metode kombinuju ideje iz prethodnih, srodnih istraživanja: SVM [6, 8], konvolutivne mreže [10] kao inicijativno isprobane: Logistička regresija, *Random Forest* i rekurentne neuronske mreže.

Rezultati pokazuju da je detekcija sarkazma u tekstu moguća, ali da je ipak potrebno refiniranje modela kako bi njihova primena u kombinaciji sa modelima za analizu sentimenta doprinela sveopštem cilju. Predloženi su mogući dalji koraci razvoja u ovom smeru, koji bi omogućili povećanje tačnosti modela za detekciju sarkazma.

6. LITERATURA

[1] Sarcastic sentiment detection in tweets streamed in real time: a big data approach, K.Bharti, B.Vachha, R.K.Pradhan, K.S.Babu, S.K.Jena

[2] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[3] https://scikit-learn.org/stable/modules/svm.html

[4] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[5] Convolutional neural networks for sentence, Kim, Yoon, 2014

[6] Detecting Sarcasm in Text, Peng, Chun-Che; Lakis, Mohammad, Pan, Jan Wei.

[7] http://www.thesarcasmdetector.com/

[8] Sarcastic or not: Word embeddings to predict the literal or sarcastic meaning of words, Ghosh, Debanjan; Guo, Weiwei; Muresan, Smaranda.

[9] https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa

[10] A deeper look into sarcastic tweets using deep convolutional neural networks, Poria, Soujanya

[11] https://nlp.stanford.edu/projects/glove/

[12] https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit

Kratka biografija:

Sara Perić rođena je 1996. godine u Loznici. Osnovne akademske studije završila je 2018. godine na Fakultetu tehničkih nauka, na kom brani i master rad 2019. godine iz oblasti Elektrotehnike i računarstva – Softversko inženjerstvo i informacione tehnologije.
kontakt: sara.peric013@gmail.com

Зборник радова Факултета техничких наука, Нови Сад

UDK: 004.4
DOI: https://doi.org/10.24867/07BE26Jeftic

РАЗВОЈ БИБЛИОТЕКЕ ЗА ИНДУСТРИЈСКИ КОМУНИКАЦИОНИ ПРОТОКОЛ *SERIES V*

DEVELOPMENT OF THE LIBRARY FOR THE INDUSTRIAL COMMUNICATION PROTOCOL SERIES V

Филип Јефтић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *У овом раду представљено је једно рјешење програмске подршке за управљање Slave страном у SCADA системима употребом Series V протокола. Поред описаног концепта рјешења, у раду су описане основе SCADA система, комуникационих и индустријских протокола као и самог Series V протокола.*

Кључне ријечи: SCADA, RTU, SERIES V

Abstract – *This paper presents a software solution for Slave side managing in SCADA systems using Series V protocol. In additional to concept of the solution, there are described theoretical concepts of SCADA systems, communication and industrial protocols as same as the Series V protocol.*

Keyords: SCADA, RTU, SERIES V

1. УВОД

Данас су људи, индустрија и велике корпорације постали зависни од електричне енергије, то јесте од њеног континуалног напајања. И најмањи испади у електро системима могу довести до штете која се мјери милионима евра. С тога је основни и главни циљ електро инжењера да учине тај систем што робуснијим, поузданијим и сигурнијим. Међутим, ту се крије велика одговорност и обавеза, јер је у питању огромна количина података које је потребно обрадити у реалном времену. Ријеч је о десетинама милиона тачака. Данас се слободно може рећи да податак представља главни и најважнији ресурс и с тога је неопходно правилно руковати са њим. Тако долазимо до термина *SCADA* (*Supervisory Control And Data Acquisition*) система и индустријских комуникационих протокола.

Идеја овог рада је приједлог рјешења за имплементацију програмске библиотеке на примјеру *Series V* индустријског комуникационог протокола. Циљ је да се одради и детаљно опише сам поступак имплементације библиотеке. Поред описане имплементације и концепта рјешења, у раду су описане теоријске основе *SCADA* система, комуникационих и индустријских протокола.

НАПОМЕНА:

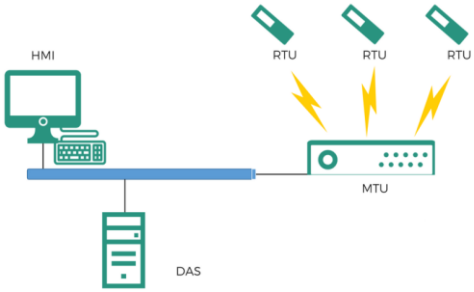
Овај рад проистекао је из мастер рада чији ментор је био др Бранислав Атлагић, доцент.

2. ТЕОРИЈСКЕ ОСНОВЕ

2.1 *SCADA* систем

Аквизиционо управљачки системи (*АУС*) су базирани на употреби рачунара и дигиталних рачунарских компоненти. Њихов основни циљ је обезбјеђење ефикасног надзора и управљања над произвољним физичким процесом. Овакви системи, специфичне намјене и структуре, у литератури се најчешће означавају термином *SCADA* (*Supervisory Control And Data Acquisition*). Фундаментални захтјеви које аквизиционо управљачки систем опште намјене мора испунити су:

- Рад у реалном времену,
- Дистрибуција рачунарских ресурса у оквиру аквизиционо управљачког система,
- Постизање максималне поузданости и расположивости.



Слика 2.1.1 Архитектура SCADA мреже

MTU (Master Terminal Units) или водећа процесна јединица у *SCADA* систему представља уређај који издаје команде над *RTU* уређајима, прикупља захтјеване податке, складишти информације, процесира информације и приказује их у форми слика, графова и табела на корисничком приказном уређају (*HMI*) и на тај начин помаже да се лакше доносе одлуке (слика 2.1.1). Ово је задатак *MTU-a* лоцираног у контролном центру. *DAS (Direct-attached storage)* представља дигитално складиште података прикључено на рачунар који му приступа [2].

Циљеви које треба остварити у току пројектовања и извођења аквизиционо управљачких система су сљедећи [1]:

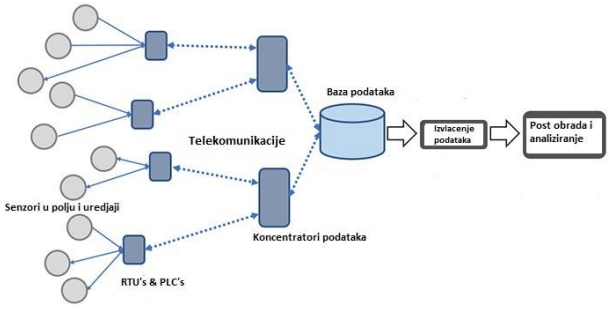
- *Смањење трошкова производње.* Оно се остварује на два начина. Први је везан за мањи обим потребне радне снаге, смањење трошкова превоза, повећање квалитета производње и сл. Други аспект се огледа у повећаном степену сигурности извршења технолошких процеса који

често могу бити екстремно опасни по околину или животе људи.

- *Расположивост и интегритет система*. АУС мора обезбиједити наставак рада и у случају отказа појединих компоненти. Интегритет (конзистентност) података и укупно функционисање не смије бити угрожено.
- *Флексибилност и проширивост система*. У току употребе аквизиционо управљачког система често су потребне измјене његове конфигурације. Оне су најчешће условљене промјенама у производном процесу који се надзире.
- *Поузданост система*. Поузданост АУС система условљена је поузданошћу његових саставних компоненти.
- *Перформанса система*. Генерално, перформанса се дефинише помоћу времена одзива и пропусности. Вријеме одзива се може смањити нарочито уколико се главнина обраде врши локално, без потребе за чекање управљачког сигнала од централне станице. *Пропусност* се односи на обим података који се могу пренијети и обрадити у оквиру аквизиционо управљачког система.

2.1.1 *Polling* у *SCADA* системима

Када причамо о *SCADA* системима, често се провлачи појам *Polling-a* (*слика 2.1.1*). Он представља процес којим *SCADA* прикупља податке. Порука се пошаље до *RTU* уређаја и чека се на одговор. Тај уређај посједује везе које га вежу за физички свијет, у којем се врше мјерења напона, струје и фазе. Одговор који се шаље назад су специфичне вриједности захтјеване преко *polling-a* [6].



Слика 2.1.1.1 Polling података

Poll захтјев може да садржи генерални захтјев као на примјер читање вриједности напона, или једноставно може бити фокусиран на специфична мјерења. Један примјер захтјева може бити рецимо прибављање температуре трансформатора. Тај захтјев се шаље када рецимо желимо да знамо да ли се одређени трансформатор прегријао [6].

Овај процес *polling-a* се може вршити свако пар секунди, минута, сати, дана, мјесеци или година. Процес је зависан од типа опреме, њене улоге у систему и ризика који та опрема носи са собом [6].

Међутим, увијек постоји могућност појаве непредвиђене ситуације, што је уједно и битна карактеристика критичних система. На примјер, постоји *polling* процес који се извршава свако 30 минута, а рецимо у 15 минуту се десио неки испад, односно неко прекорачење. Тај дио опреме који је измјерио прекорачење ће да обавијести *RTU* уређај који ће затим да изда захтјев за *poll* који се зове *unsolicited poll response* (више о овоме у поглављу *2.3.1*). Овај захтјев ће се затим послати кроз *SCADA* систем, који ће препознати о којем типу захтјева је ријеч. Ово је јако чест процес [6].

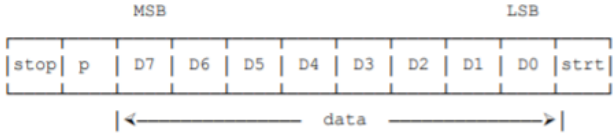
3. *SERIES V* ПРОТОКОЛ

Series V SCADA Communication protocol је асинхрони *byte* оријентисани протокол чији је примарни циљ командовање и аквизиција над *RTU* уређајима у пољу, користећи исти комуникациони канал. Дизајниран је тако да се повеже директно на рачунарске комуникационе портове. Може се користити у *point-to-point* или *multi-drop* конфигурацији. Комуникација између двије компоненте се може остварити у *half* или *full-duplex* варијанти [4][5].

Сваки вид комуникације је иницран од стране водеће јединице (*MTU*). Удаљена процесна јединица (*slave* страна) не може да иницира никакав вид размјене информација са водећом јединицом, нити може да директно адресира или комуницира са другом удаљеном процесном јединицом. Она ће да врати одговарајући испроцесирани одговор за сваку валидну поруку која је стигла и која је адресирана на дату процесну јединицу. Ово је небалансирани режим комуникације, о коме је било више ријечи у претходном поглављу (2.3.1). Једини изузетак за ово је тип поруке која се шаље свакој станици (*broadcast message*) и у том случају ниједна удаљена јединица не враћа одговор. Приликом сваког пријема поруке се врши провјера исправности послатих података. За то се користе два уобичајена и на широко коришћена сигурносна механизма: *CRC* и *LRC*. Уколико неки од *byte*-ова није валидан, удаљена процесна јединица ће игнорисати послату поруку и никакав вид акције неће бити подузет [5].

3.1 Структура карактера

Подаци се преносе у стандардном 10-обитном или 11-обитном асинхроном бајт оријентисаном формату. Сваки од бајтова се састоји од почетног бита (*start bit*), 8 бита који представљају саме податке, опциони бит за паритет и крајњег бита (*stop bit*) [4].



Слика 3.1.1 Структура једног бајта поруке

Кратак опис садржаја једног бајта: [4]

- *strt* - почетни бит, који означава почетак бајта,
- *stop* - крајњи бит, који означава крај бајта,
- *D0-D7* - подаци, гдје је *D0* најмање значајан бит (*LSB*), а *D7* најзначајнији бит (*MSB*).
- *p* - бит паритета

3.3 *Series V* поруке

У суштини, све поруке унутар *Series V* индустријског протокола се могу подијелити у два основна типа: захтјеви за подацима и контролни захтјеви. Први тип захтјева податке у виду вриједности са *RTU* уређаја. Ови подаци могу бити дискренти, аналогни, акумулатор, израчунате варијабле, параметри удаљене једи-

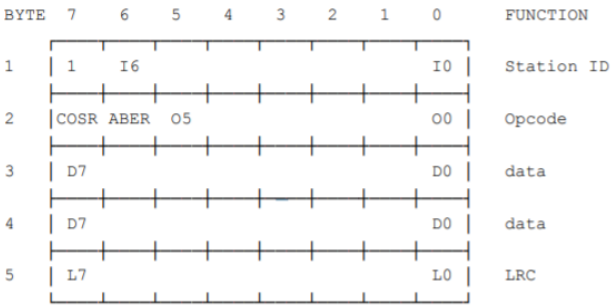
нице, статус *RTU* уређаја, аналогни излази и дискретни излази.
Други тип захтјева је дефинисан као било која порука упућена од стране водеће процесне јединице која захтјева промјену стања неког од уређаја у пољу или модификацију интерног стања удаљене процесне јединице [5].

3.3.1 Структура поруке

Ради лакоће приказа, почетни, крајњи и бит паритета ће бити изостављени у наредним илустрацијама. За више информација о њима, погледајте поглавље 3.1.
Такође, у наставку је приказана структура порука са *LRC* сигурносним механизмом, што значи да је величина поруке 5 бајта.
Све поруке које се размјењују између *MTU* и *RTU* имају неку од основних структура наведених у наставку [4].

3.3.1.1 *Master-to-Remote* порука

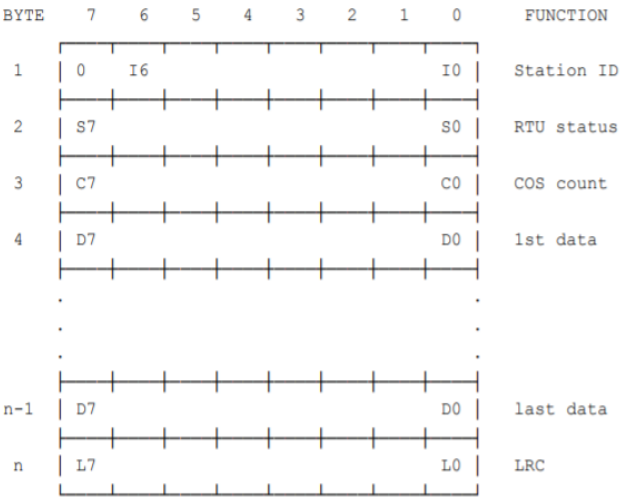
Структура поруке је илустрована на слици 3.3.1.1.1.



Слика 3.3.1.1.1 Структура поруке од MTU до RTU

3.3.1.1 *Remote-to-Master* порука

Структура овог типа поруке је приказана на слици 3.3.1.2.1.

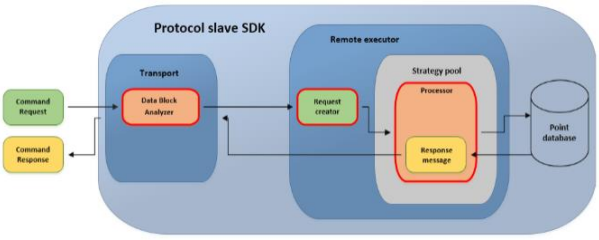


Слика 3.3.1.2.1 Структура поруке од RTU до MTU

4. КОНЦЕПТ РЈЕШЕЊА

Тема овог рада јесте да се реализује једно програмско рјешење библиотеке на примјеру индустријског комуникационог протокола *Series V*.

Црвеним оквиром је приказана примјена већ поменуте библиотеке над *Protocol Slave SDK (слика 4.1)*.



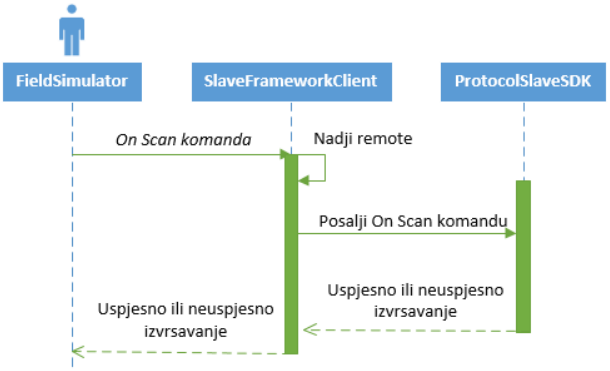
Слика 4.1 Примјена библиотеке на Protocol Slave SDK

4.1 *Protocol slave SDK*

На слици (*слика 4.1*) приказани *SDK* служи као подршка за рад са *slave* страном. У ситуацијама када програмери нису у могућности да тестирају функционалност свог софтвера над стварним уређајем у пољу, јавља се потреба за симулационим окружењем. Управо је то улога већ поменутог *Protocol slave SDK.* Међутим, оно што га раздваја од стварног уређаја је могућност манипулације и симулирања понашања свих индустријских протокола (или барем велике већине).

4.2 *Field Simulator*

За потпуни кориснички доживљај неопходан је графички приказ који омогућава директну манипулацију над *Protocol slave SDK* преко *SlaveFrameworkClient-a*.
Неке од основних функционалности које пружа *FieldSimulator* су могућност *scan on/off* свих *RTU* уређаја, канала, линкова, као и директно мијењање вриједности тачака у пољу. Пружа увид у вријеме (*TimeStamp*) посљедње очитане вриједности тачке.
Секвенцијални дијаграм (*слика 4.2.1*) приказује ток размјене порука приликом извршавања *on scan* команде преко *FieldSimulatora.*



Слика 4.2.1 Извршавање On Scan команде

4.3 *Slave Framework Client*

SlaveFrameworkClient у свој овој причи служи као као неки омотач (*wrapper*) око *ProtocolSlaveSDK* и пружа неке додатне могућности и мапирања података.
Један примјер је интерна колекција *remote* уређаја која служи за мапирање пристиглог идентификатора уређаја са конкретним уређајем.
Оно што је од велике важности је почетна иницијализација комплетног *ProtocolSlaveSDK*. Под тим се мисли на иницијализацију процесора, локаних модела *RTU* уређаја, типова захтјева, базе података и остало.

5. ИМПЛЕМЕНТАЦИЈА РЈЕШЕЊА

5.1 *Series V Data Block Analyzer*

Преко мреже стиже гомила сирових података у виду низа бајтова. Да би се ти подаци могли искористити, јавила се потреба за неким механизмом који би те податке некако прихватио, извалидирао и проследио на даљу обраду. Управо је то улога већ поменутог модула.

5.2 *Series V Request Creator*

За сваки код операције (карактеристичан за сваки протокол) се имплементира посебан тип захтјева. Сваки појединачни захтјев се зарегиструје преко овог модула у *Series5SlaveFrameworkClient*-у. Регистрација се врши унутар локалне колекције (*Dictionary*) према коду операције.

5.2 *Processors, Requests, Responses*

Процесори су организовани тако да насљеђују базни процесор који даље насљеђује одређени *interface.* Главни задатак процесора је да одради основну логику везану за процесирање пристигле команде.

Захтјеви (*requests*) су такође организовани на начин да насљеђују базни захтјев који даље насљеђује одговарајући *interface* Пошто се приликом имплементације водило рачуна о перформансама софтвера, тако се у овом случају искористила предност секвенцијалног извршавања команде, па је извршена уштеда у меморији на начин да се не праве стално нови захтеви, већ се врше промјене над почетно креираним захтјевом.

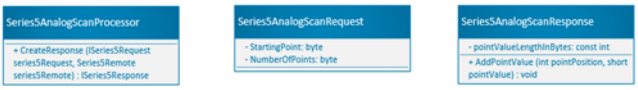
Одговори (*responses*) су реализовани преко базног одговора који даље насљеђује *interface*, али без могућности измјене вриједности над постојећим одговорима. За сваку пристиглу команду се формира посебан одговор.

Сваки захтјев, одговор и процесор из наредних поглавља насљеђују један од горе неведених класних модела. Тренутно имплементационо рјешење пружа подршку за сљедеће команде:

- *Analog scan*
- *Status point scan*
- *RTU status clear*
- *Analog change count*
- *Analog change dump*
- *Change of state queue dump*
- *Control select*
- *Control execute*
- *SOE time sync*

5.3.1 *Analog scan*

Примјер класног модела за једну од команди је приказан на слици 5.3.1.1. *Analog scan* команда враћа аналогне вриједности за захтјевани број тачака са *RTU* уређаја. Аналогни улази су нумерисани полазећи од тачке са кординатом 0. Аналогне вриједности су приказане као 12-обитне сирове вриједности у другом комплементу или као неозначене бинарне, зависно од тога како је *RTU* уређај исконфигурисан.



Слика 5.3.1.1 Класни модел за analog scan

Почетна захтјевана тачка мора бити у опсегу исконфигурисаних тачака у систему или другим ријечима, мора да постоји. Такође, крајња захтјевана тачка мора да буде мања од највеће исконфигурисане. Ако неки од претходних услова није задовољен, постављају се одговарајући *flag*-ови.

6. ЗАКЉУЧАК

У овом раду је представљено програмско рјешење библиотеке за *ProtocolSlaveSDK*, на примјеру *Series V* индустријског комуникационог протокола.

За реализацију рада је било неопходно детаљно упознавање са индустријским протоколима, као и са архитектуром и понашањем *Series V* протокола. Идеја је била да се дође до рјешења које ће бити максимално оптимизовано и перформантно, те ће на тај начин омогућити обраду огромне количине команди у кратком временском периоду.

7. ЛИТЕРАТУРА

[1] Софтвер са критичним одзивом, 2015, Бранислав Атлагић

[2] http://electricalquestionsguide.blogspot.com/2011/06/master-terminal-units-mtu-in-scada.html

[4] Series V Communication protocol, © Copyright 1992 by Valmet Automation

[5] Tejas protocol emulation, www.scribd.com

[6] https://www.taitradioacademy.com/topic/scada-and-polling-1/

Кратка биографија

Филип Јефтић рођен је 16.04.1995 године на Илици. Завршио је Гимназију “Јован Дучић” у Требињу 2014 године. Исте године је уписао основне академске студије на Факултету техничких наука у Новом Саду. Дипломски рад из области Електротехника и рачунарство – Примењени софтверски инжењеринг одбранио је 2018. године. Испунио је све обавезе и положио све испите предвиђене студијским програмом.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE28Vukasinovic

ARHITEKTURA VEB-BAZIRANOG SISTEMA ZA NADZOR I UPRAVLJANJE UREĐAJIMA

ARCHITECTURE OF A WEB-BASED SYSTEM FOR DEVICE MONITOR AND CONTROL

Nikola Vukašinović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Ovaj rad se bavi problematikom kreiranja platforme veb-baziranog sistema čija je primarna uloga udaljen nadzor i kontrola uređaja. Sistemi za upravljanje i kontrolu karakteriše veliki broj članova, svaki sa zasebnim zadacima koji međusobnom komunikacijom dovode do realizovanja namenske funkcionalnosti rešenja. Analiza se prevashodno svodi na opisivanje arhitekture sistema koja uključuje opisivanje svakog pojedinačnog podsistema. Posebna pažnja poklonjena je komunikacionom aspektu podsistema sa akcentom na izbor protokola mreže, formatom i načinom prenosa poruka. Drugi aspekt analize podrazumeva opis implementacije sistema, gde je predstavljana realizacija svakog pojedinačnog člana u okviru korištene programske paradigme.*

Ključne reči: *Sistemi za kontrolu i upravljanje, pametni domovi, veb sistemi kontrole*

Abstract – *The goal of this paper is to describe a system of a web-based platform for device monitoring and control. Systems of this nature contain a large number of components each with its own task who communicate with each other in other for the solution to function properly. The primary goal is to fully describe the system layout and architecture by analysing each component individually. Special attention is given to cross component communication by analysing different network protocols, specifying a message format and the functionality of sending those messages. The second part of the analysis is focused on describing the way the system as a whole is implemented through its components and the programming languages used for their individual implementation.*

Keywords: *Surveillance and control systems, smart home, control web systems*

1. UVOD

Koncept kontrole i automatizacije kućnih uređaja je relativno star pojam u odnosu na životni vek informacionog doba. Godine 1975-te javlja se *X10* kao prvi standardizovani protokol čija je namena udaljenog upravljanja kućnih aparata realizovana slanjem kratkih radio-signala kroz postojeću električnu arhitekturu.

Mane ovog protokola manifestuju se u njegovoj nepouzdanosti i nemogućnosti dvosmerne komunikacije. Sama arhitektura postojećih električnih instalacija nije savršeno kompatibilna za prenos radio-signala. Pojavljivanjem povremenih šumova uzrokovana je nepouzdanost primljenog signala. Jednosmerna komunikacija onemogućila je bilo kakvu mogućnost slanja povratnih informacija od uređaja ka korisniku čime se uveliko ograničava sfera korišćenja ovakvih sistema.

Razvojem interneta pojavljuje se potreba takozvanog udaljenog (*remote access*) pristupa. Koncept po kome se uređaji u prostorijama mogu nadgledati i upravljati sa relativnom sigurnošću iz bilo koje tačke na planeti u realnom vremenu. Uporedno sa razvojem interneta, razvijaju se novi, i unapređuju postojeći komunikacioni protokoli koji omogućavaju obostranu trenutnu komunikaciju između uređaja u relativnoj blizini.

Svi prevashodno definisani koncepti se grupišu u jedan sistem za kontrolu i upravljanje uređajima. Svrha ovih sistema je omogućavanje krajnjim korisnicima da na lak, brz i intuitivan način kontrolišu, upravljaju i automatizuju svoje prostorije svrsishodno ličnim preferencama.

Ciljevi analize se prvenstveno odnose na dva glavna aspekta. Prvi, predstavlja opis arhitekture, analizu funkcionalnih operacija članova sistema i načine njihove međusobne komunikacije, kao i načine dodavanja i brisanja uređaja. Drugi se odnosi na predstavljanje lične implementacije jednog ovakvog sistema i opisivanjem svakog pojedinačnog člana u okviru programske platforme njegove implementacije. Takođe, potrebno je analizirati kretanje informacija od uređaja do krajnjeg korisnika u oba smera.

2. ARHITEKTRA SISTEMA

Zadaci sistema ogledaju se u:

- Definisanju veb-bazirane platforme preko koje svaki korisnik može nadgledati i upravljati svoje uređaje.
- Omogućiti korisnicima kreiranje, brisanje, dodavanje i modifikaciju uređaja za prikupljanje podataka jednog kućnog sistema, kao i njihovo grupisanje u jednu jedinstvenu tačku, u daljem nastavku čvorište.
- Mogućnosti opsluživanja što većeg broja korisnika u datom momentu.
- Prihvatanju numeričkih podataka poslatih od strane mernih senzora ka čvorištu nezavisno od frekvencije slanja. Omogućiti uporedno slanje povratnih informacije ka senzorima.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Branko Milosavljević, red. prof.

- Definisanje logike za prosleđivanje podataka aplikativnom serveru.
- Prihvatanje podataka od strane servera i prosleđivanje istih aplikaciji korisničkog interfejsa.
- Definisanje valjanog i intuitivnog grafičkog korisničkog interfejsa koji omogućava nadgledanje i slanje komandi definisanim uređajima preko ostvarenih komunikacionih veza.

Shodno predstavljenim zadacima arhitektura je izgrađena sa ciljem da u što efikasnijoj meri omogući ispunjenje funkcionalnih zahteva.

Takođe, imajući u vidu da je analizirani sistem prvenstveno veb-baziran, skalabilnost predstavlja još jedan ključni zahtev koji treba imati na umu. Ključne karakteristike projektovanja arhitekture su:

1. *Nezavisnost komponenti sistema* – koja govori o osobini po kojoj glavne funkcionalnosti jedne komponente minimalno moraju da zavise od ostalih u sistemu.
2. *Oslobađanje resursa na serveru* – koji se odnosi na oslobađanje aplikativnog servera od potrebe poznavanja korisničke konfiguracije mernih i kontrolnih uređaja.
3. *Definisanje jasnih komunikacionih veza kao i univerzalnih poruka za komunikaciju* – koja je produkt nezavisnosti komponenti
4. *Razdvajanje logike prikupljanja podataka i kontrole od logike predstavljanja istih* – po svojoj prirodi arhitektura razlikuje korisničku i serversku konfiguraciju.
5. *Mogućnost sistema da podrži što veći broj merno-upravljačkih uređaja* – čime se povećavaju konfiguracione mogućnosti.
6. *Jednostavno proširivanje funkcionalnosti sistema* – gde je platforma izgrađena sa ciljem da lako i jednostavno obezbedi mogućnost proširivanja funkcionalnosti sa akcentom na štednji resursa i novca.

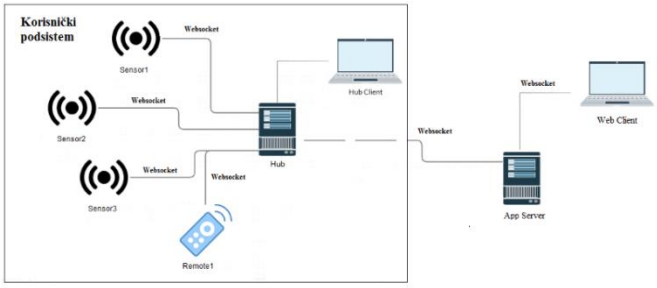
Imajući u vidu zadatke i karakteristike projektovanja arhitekture ceo sistem za upravljanje i kontrolu može se podeliti na dva strogo izolovana podsistema: *korisnički i serverski podsistem*.

Korisnički podsistem bavi se ispunjenjem konfiguraciono-operacionih funkcionalnosti vezanih za jednog korisnika. Definisanje konfiguracije merno-upravljačkih senzora, prikupljanje mernih podataka i prosleđivanje istih aplikativnom serveru predstavljaju glavne funkcionalne zahteve koji podsistem treba da ispuni. Bitno je istaći da svaki korisnik sadrži svoju instancu podsistema, odnosno jedan podsistem vezan je za tačno jednog korisnika platforme.

Serverski podsistem služi kao mesto pristupa krajnjeg korisnika sistemu. Funkcionalni zadaci serverskog podsistema odnose se na prikupljanje i prenos podataka od različitih korisničkih podsistema, grafičko prikazivanje izmerenih podataka i prenos korisničkog unosa do adekvatnog podsistema.

Članovi korisničkog sistema predstavljaju:

- Upravljački i merni uređaji
- Čvorište podsistema
- Aplikacija korisničkog upravljanja



Slika 1. *Celokupna arhitektura sistema*

Upravljački i merni uređaji predstavljaju glavnu operativnu jedinicu celokupnog sistema. Primarna uloga ogleda se u prikupljanju podataka o trenutnom stanju prostorija različite prirode. Sam broj, fizička lokacija i položaj ovih uređaja u potpunosti su prepušteni željama krajnjih korisnika. Uređaji prikupljaju podatke čija je priroda ograničena pravilima sistema dok, u konkretnom rešenju, dozvoljeni su podaci o: temperaturi, nivou vode, dimu, količini potrošene energije, gasovima, kretanju i stanju aparata za kontrolu temperature. Potrebno je napomenuti da po šestoj stavci karakteristika arhitekture, ovu listu je moguće proširiti. Da bi se uređaj karakterisao kao deo podsistema on mora uspostaviti vezu sa čvorištem i imati mogućnost slanja podataka dozvoljene prirode preko uspostavljanje veze u numeričkom formatu. Uređaj se smatra operativno-funkcionalnim u koliko je registrovan na čvorište korisničkog podsistema. Svaki uređaj sadrži svoj identifikacioni broj čime se osigurava njegovo jedinstveno prepoznavanje od strane aplikacije čvorišta, unutar podsistema.

Čvorište predstavlja glavnu logičko-funkcionalnu jedinicu korisničkog podsistema. Definisano kao centralni deo podsistema, prvenstveno služi kao tačka uspostavljanja veza sa svim uređajima kao i aplikativnim serverom. Svaki korisnički podsistem sadrži tačno jedno čvorište. Zadatak ostvarivanja veze čvorišta i uređaja je ključan preduslov funkcionisanja korisničkog podsistema. S obzirom na prirodu samog podsistema potrebno je uspostaviti veze između potencijalno velikog broja članova koji su fizički u relativno neposrednoj blizini. Adekvatno rešenje zahteva korišćenje bežične komunikacije, dok je sam tip protokola mrežne komunikacije i dalje predmet diskusije i razlikuje se od sistema do sistema. Konačni izbor mrežnog protokola korisničkog podsistema u potpunosti zavisi od namene, okruženja i bitnih karakteristika projektovanja.

Protokoli čije prednosti i mane predstavljaju domen analize su: *Wifi*, *Zigbee*, *Z-wave i Bluetooth*. Prednosti *Wifi* mreže prvenstveno se ogledaju u njenoj dostupnosti i širokoj primeni. Oslanjanjem na postojeću korisničku arhitekturu oslobađamo se od potrebe uvođenja nove u cilju realizovanja komunikacije.

Mane ovog pristupa prvenstveno se ogledaju u njenoj ceni, korišćenju frekvencijskog opsega oko 2.4 GHz (opseg se mnogo koristi i mogu da se pojave smetnje u komunikaciji) kao i mogućom potrebom ulaganja u kvalitetnije mrežne uređaje s obzirom da komercijalni kućni uređaji često nemaju dovoljno snage i dometa da se nose mrežnim potrebama koje korisnički podsistem zahteva.

Zigbee i Z-wave predstavljaju komunikacione protokole razvijene sa ciljem da zadovolje potrebe sistema za upravljanje i kontrolu [1]. *Zigbee* odlikuje mrežna moć sa mogućnošću uvezivanja velikog broja uređaja u jednom momentu.

Takođe, uređaji koji komuniciraju ovim protokolima odlikuju se relativno malom potrošnjom energije. Prednost *Z-wave* protokola proizilazi iz upotrebe frekvencijskog opsega (800-900Mhz) što ga čini imunim na potencijalne smetnje prouzrokovane ostalim signalima i samim tim, najpouzdanijim rešenjem. Korišćenje *Bluetooth* protokola za potrebe korisničkog podsistema ograničava se na sisteme sa relativno malim brojem uređaja i malim dometom. Odlikuju ga mala potrošnja energije kao i mogućnost svakog uređaja da ima primopredajne karakteristike koje proizilaze iz same definicije mreže.

Nepouzdanost signala i mali domet predstavljaju glavne mane korišćenja ovog rešenja čime se zahteva dalje razvijanje tehnologije kako bi ona pronašla širu primenu u domenu sistema za upravljanje i kontrolu.

Kada je reč o komunikaciji između čvorišta i aplikativnog servera glavni aspekt predstavlja faktor trenutnog (*real time)* prenosa poruka. Imajući na umu da se komunikacija obavlja preko interneta, standardni *HTTP* protokol nije adekvatan za ostvarivanje ovakve komunikacije. Limitirajući faktor predstavlja zahtev-odgovor karakteristike *HTTP* protokola gde se pri svakom slanju poruke mora prvobitno poslati zahtev i čekati odgovor na isti čime se veza zatvara i svako novo slanje zahteva kreiranje novog zahteva što vremenski nije prihvatljivo.

Takođe, sam broj transakcija koji se otvaraju, zatvaraju i izvršavaju u jednoj jedinici vremena od strane velikog broja korisnika prouzrokovala bi smetnje u funkcionisanju aplikativnog servera. Rešenje je realizovano uvođenjem *Websocket* konekcije između čvorišta i aplikativnog servera.

Za razliku od *HTTP-a*, *Websocket* se zasniva na uspostavljanju stalne veze između dve partije omogućavajući slobodan i trenutan protok informacija. Nakon ostvarivanja, veza dozvoljava klijentu i serveru slanje podataka kroz otvorenu konekciju u vidu poruka bez prethodne potrebe druge strane da te informacije zatraži i bez dodatnih informacija kao što su zaglavlja akcentujući pojam trenutne komunikacije. *Websocket* protokol izgrađen je preko *HTTP-a* obzirom da se inicijalni zahtev za uspostavljanje veze (*handshake*) šalje preko *HTTP* zahteva. Ova veza takođe doprinosi većoj skalabilnosti aplikativnog servera koji otvaranjem samo jedne veze sa čvorištem oslobađa resurse čime se omogućava istovremeno opsluživanje većeg broja korisnika.

Za uspešno opisivanje logike prenosa podataka potrebno je obratiti pažnju na vremenski faktor. Merni uređaji šalju izmerene podatke čvorištu korišćenjem definisane mrežne arhitekture različitim frekvencijama.

Takođe, podaci od čvorišta do aplikativnog servera šalju se unapred definisanim intervalima slanja. Pravilno funkcionisanje čvorišta zahteva definisanje logike za prikupljanje, skladištenje, računanje proseka i slanje datih merenja. Problematika slanja podataka aplikativnom serveru može se svesti na problem „*jedan senzor, jedan broj*“, kojom se definiše potreba da svaka vrednost senzora bude predstavljanja tačno jednim mernim rezultatom nasuprot mogućnosti senzora da u vremenskom periodu od prethodnog do novog slanja na čvorište pošalje jedno, više ili ni jedno merenje. Slučaj gde u periodu od prethodnog do sledećeg slanja nije pristiglo ni jedno merenje realizuje se prosleđivanjem prethodno izmerene vrednosti ponovo, sa određivanjem praga tolerancije.

Slanje nove vrednosti podrazumeva računanje centralne tendencije skupa merenja pristiglih od trenutka prethodnog slanja. Statističkim metodama zaključeno je da računanje medijane skupa podataka predstavlja najbolje rešenje računanja centralne tendencije s obzirom da je ova metoda imuna na retke nepravilnosti u merenju što je neretka pojava u funkcionisanju svih senzora merenja [2].

Klijentska aplikacija korisničkog podsistema predstavlja tačku interakcije korisnika sa konfiguracijom podsistema i ogleda se u pružanju korisničkog interfejsa i omogućavanju funkcija za dodavanje i brisanje registrovanih uređaja korisničkog podsistema, kao i uspostavljanje i raskidanje konekcije sa aplikativnim serverom.

Model serverskog podsistema svodi se na klijent-server arhitekturu. Primarna uloga aplikativnog servera ogleda se u omogućavanju funkcionalnosti prijave na sistem, uspostavljanje veza sa korisničkim podsistemima i prenosom informacije istih do adekvatnih klijenata ulogovanog korisnika.

Veb - klijentska aplikacija služi kao mesto interakcije korisnika sa funkcionalnostima celokupnog sistema. S obzirom na prirodu sistema, glavni akcenat projektovanja veb-klijenta odnosi se na pružanje intuitivnog grafičkog interfejsa za prikazivanje pristiglih numeričkih podataka i davanjem mogućnosti korisničkog unosa definisanjem adekvatnih formi.

2. IMPLEMENTACIJA SISTEMA

Opis implementacije sistema za kontrolu i upravljanje uređajima odnosi se na način realizacije funkcionalnih karakteristika svakog od člana sistema u okviru određene programske paradigme prema definisanim pravilima arhitekture. *Websocket* veza povlači sa sobom definisanje protokola za organizaciju poruka čiju potrebu u analiziranom rešenju ispunjava *STOMP* protokol [3]. *STOMP* uvodi novi sloj organizacije destinacija poruka za slanje i primanje po putanjama, dok je funkcionalnost realizovana preko različitih komandi.

2.1 Merni i upravljački uređaji

Implementacija merno-upravljačkih uređaja realizovana je korišćenjem *Arduino* platforme. Platforma se odnosi na hardversku mikro-kontroler ploču sa analognim i digitalnim ulazima i izlazima koja služi za integrisanje različitih malih električnih uređaja. Konkretno, merni uređaj predstavlja *DS18B20* temperaturni senzor dok kontrolni uređaj je predstavlja *IR* lampica koja kontroliše klima uređaj prosleđivanjem adekvatnih komandi. Veza između ploče i računara ostvarena je preko komunikacionog porta na koji se šalju rezultati merenja i komande. Aplikacija senzora realizovana je korišćenjem *SPRING* framework-a preko *JAVA* programske platforme. Glavni zadaci ove aplikacije ogleda se u čitanju komunikacionog

porta i prosleđivanju izmerenih vrednosti na čvorište kao i prosleđivanju komandi primljenih od čvorišta na komunikacioni port koji kasnije dolaze do ploče.

2.2 Čvorište korisničkog podsistema

Analizirani sistem definiše čvorište kao zasebnu server aplikaciju razvijenu u *JAVA* programskoj paradigmi korišćenjem *SPRING* framework-a. Registracija uređaja na čvorište podsistema realizuje se upisivanjem informacija o istom u internu bazu podataka čvorišta. Svaki uređaj definisan je informacijama identifikatora, naziva i tipa koji opisuje prirodu poslatih podataka iz skupa mogućih vrednosti. Manipulisanje konfiguracijom korisničkog podsistema podrazumeva kreiranje zasebnog *API*-ja koji, zajedno sa klijentom korisničkog podsistema, realizuje date funkcionalnosti.

Povezivanje uređaja na čvorište realizuje se iniciranjem sesije između dve partije. Model sesije definisan je informacijama uređaja, nizom pristiglih merenja, informaciju o poslednje poslatoj izračunatoj vrednosti tendencije, trenutnom pragu tolerancije i logičkoj promenjivoj koja govori o aktivnosti samog uređaja.

```
public class ComponentSession {

    private Component component;
    private List<Double> stack = new
ArrayList<Double>();
    private double lastSent;
    private boolean sent = false;
    private int counter = 0;
```

Kod 1. *Model klase sesije uređaja i čvorišta*

Standardizovana poruka za prenos mernih informacija predstavlja nužan preduslov valjanog rešenja. Bitan faktor postizanja ove standardizacije predstavlja usklađenost date poruke se mapom aktivnih sesija. Kreiranje modela poruke o stanju korisničkog podsistema ima za cilj da pruža realno stanje izmerenih podataka organizovanih na način na koji olakšava njihovu kasniju navigaciju i interpretiranje. Format poruke je definisan kao niz čija je dužina jednaka brojčanoj količini tipa senzora čija se merenja šalju za dati vremenski interval, dok svaki član niza sadrži podatke o merenjima senzora datog tipa. Pravila slanja podataka definišu pojavu da se u različitim vremenskim intervalima šalju podaci vezani za senzor različite prirode
Realizacija logike slanja poruka svodi se na definisanje internog brojača vremenskog intervala koji prema pravilu za slanje računa i šalje zahtevane vrednosti.

2.3 Aplikativni server

Aplikativni server funkcioniše po principu veb - servera čija se primitivna funkcionalnost ogleda u opsluživanju korisnika podacima potrebnim za nadzor i upravljanje njihovih podsistema, realizovan preko *SPRING* framework-a i *JAVA* programske platforme. Prvenstveno služi kao most koji prenosi informacije od čvorišta do odgovarajućeg klijenta preko otvorenih veza. Takođe, omogućava klijentima prijavu na sistem korišćenjem *JWT* šablona.

2.4 Veb-klijent

Veb-klijent razvijen je pomoću *Angular 7* framework-a koji se služi *Typescript* programskom paradigmom. Motivacija za korišćenje *Angular* framework-a proizilazi iz organizacije rešenja u komponente. Svaka komponenta predstavlja nezavisni deo koda sa svojom logikom funkcionisanja i zasebnim grafičkim prikazom. U analiziranom rešenju definisanje grafičkog prikaza za određeni tip merenja realizovan je kreiranjem zasebne komponente koja implementira dati prikaz.
Prednosti komponenti proizilaze iz njihove ponovne iskoristivosti, gde je pri svakom pozivu komponente data mogućnost prosleđivanja zasebnog ulaza, slično funkcijama. Predstavljenom organizacijom kao i standardizacijom poruke o stanju korisničkog podsistema osigurana je generičnost rezultata i omogućena je laka izmena logike grafičkog prikaza imajući na umu da je ona definisana samo na jednom mestu. Grafički prikaz komponenti omogućen je korišćenjem *chart.js* biblioteke.

3. ZAKLJUČAK

Prevashodni cilj analize i implementacije bio je opisivanje arhitekture jedne platforme za efikasno ispunjenje funkcionalnih zahteva predstavljenog sistema. Glavni zadatak, osim same funkcionalnosti, bio je postavljanje sistema sa čvrstom osnovom čije bi dalje unapređenje i razvijanje bilo lako i intuitivno. Faktor daljeg napretka i razvitka može se odnositi na veliki dijapazon aspekata sistema kako sa funkcionalno-implementacione tako i sa hardversko-tehničke strane. Priroda sistema omogućava individualni razvitak i proširivanje funkcionalnosti pojedinačnih članova bez kompromitovanja samog toka informacija opisanom arhitekturom.

4. LITERATURA

[1] Inovelli, https://inovelli.com/z-wave-vs-zigbee-vs-bluetooth-vs-wifi-smart-home-technology/
[2] Laerd statistics, Measures of Central Tendency https://statistics.laerd.com/statistical-guides/measures-central-tendency-mean-mode-median.php
[3] Stomp over Websocket, http://jmesnil.net/stomp-websocket/doc/
[4] Components, https://angular.io/api/core/Component

Kratka biografija:

Nikola Vukašinović rođen je u Novom Sadu 1994. god. Završio je Gimnaziju "Svetozar Marković" u Novom Sadu. Osnovne četvorogodišnje studije na Fakultetu tehničkih nauka završio je 2018 godine i iste upisuje master studije na modulu elektronsko poslovanje. Ispunio je sve obaveze i položio sve ispite predviđene studentskim programom.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE29Krajinovic

ANALIZA PROTOKOLA HTTP 2.0

ANALYSIS OF HTTP 2.0

Marko Krajinović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – *U ovom radu izvršena je analiza karakteristika komunikacionog protokola HTTP 2.0. Upoređen je ovaj protokol sa prethotnom verzijom protokola - HTTP 1.1. Analizirane su mogućnosti za primenu protokola HTTP 2.0 u serverskim radnim okvirimima Spring Boot, Microsoft ASP.NET Core i Node.js. Izvršena je analiza performansi protokola.*

Ključne reči: *Veb, HTTP*

Abstract – *In this paper the analysis of HTTP 2.0 communication protocol features has been done. Also, the comparison of this protocol with it's predecessor. Analysis of HTTP 2.0 implementation capabilities in Spring Boot, Microsoft ASP.NET Core, and Node.js server workspaces has been done as well as the analysis of the HTTP 2.0 performances.*

Keywords: *Web, HTTP*

1. KARAKTERISTIKE HTTP-A

Hypertext Transfer Protocol (HTTP) [1] je aplikativni protokol koji ne čuva stanje sesije, za distribuirane, kolaborativne, hipermedijske informacione sisteme. Predstavlja osnovu komunikacije podataka sa *World Wide Web*-a (WWW) [2].

Hipertekst podrazumeva da datoteke koje se prenose preko HTTP-a mogu sadržati reference na druge datoteke čiji će izbor izazvati dodatne zahteve za prenos. Pored veb stranica koje mogu da se serviraju, bilo koji server sadrži HTTP *daemon*, program koji je predviđen da čeka HTTP zahteve i obrađuje ih kad stignu.

HTTP funkcioniše kao protokol koji se zasniva na zahtev-odgovor mehanizmu u računarskom modelu klijent – server. Na primer, veb *browser* može biti klijent, a aplikacija koja radi na računaru koji *host*-uje veb stranicu može biti server. Klijent šalje serveru Http zahtev. Server, koji pruža resurse kao što su HTML datoteke i drugi sadržaj, ili obavlja druge funkcije u ime klijenta, vraća odgovor klijentu. Odgovor sadrži informacije o statusu završetka zahteva i može takođe da sadrži traženi sadržaj u telu http odgovora.

2. ISTORIJSKI RAZVOJ HTTP-a PRE VERZIJE 2.0

Originalni predlog za HTTP Tima Berners-Lee-a bio je osmišljen kako bi pomogao u usvajanju njegove druge ideje koja se tek rodila: *World Wide Web*. Berners Lee je 1991. izložio motivaciju za novim protokolom i nabrojao nekoliko dizajnerskih ciljeva na visokom nivou:

- funkcionalnost prenosa datoteka,
- mogućnost pretraživanja indeksirane arhive hiperteksta,
- pregovaranje o formatu, i
- sposobnost da klijenta uputi na drugi server.

Da bi se teorija dokazala na delu, izgrađen je jednostavan prototip koji je implementirao mali podskup predloženih funkcija:

- zahtev klijenta je jedan niz ASCII znakova,
- zahtev klijenta prekida se znakom za povratak prenosa,
- odgovor servera je HTML, i
- veza se prekida nakon završetka prenosa dokumenta.

Pošto je prototip bio podskup željenog protokola, nezvanično je dobio naziv HTTP 0.9 [3].

2.1. HTTP 1.0

Rastuća lista željenih mogućnosti veba i slučajeva njihove upotrebe na javnom vebu brzo su otkrili mnoga osnovna ograničenja HTTP-a 0.9. Potreban je bio protokol koji bi između ostalog mogao da:

- poslužiti više od samo hipertekstualnih dokumenata,
- pruži bogatije meta-podatke o zahtevu i odgovoru, i
- omogući pregovaranje o sadržaju.

Na osnovu ovih zahteva nastao je novi internet standard HTTP 1.0 [4]. U ovom standardu, zaglavlja zahteva i odgovora bila su čuvana kao ASCII karakteri ali sam objekat odgovora je mogao biti bilo koje vrste: HTML datoteka, obična teksualna datoteka, slika ili bilo koji drugi tip sadržaja.

Stoga je deo HTTP-a „hiperteksualni prenos“ postao pogrešan naziv nedugo nakon njegovog uvođenja. HTTP se zapravo brzo razvio u hipermedijski transport ali se originalni naziv zadržao.

2.2. HTTP 1.1

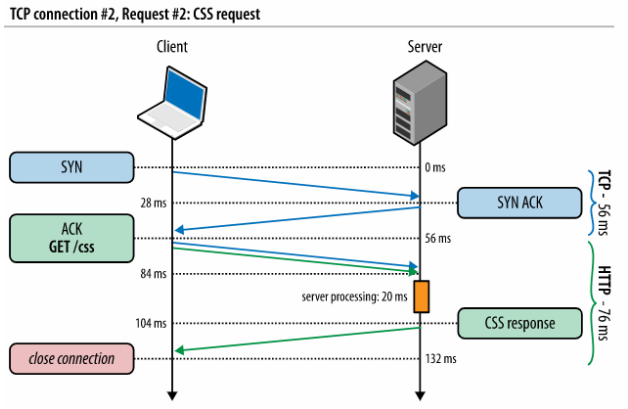
U nastavku će biti opisane prednosti ovog protokola u odnosu na prošli HTTP 1.0.

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

2.3.1 Prednosti *keep-alive* konekcije

Svaka TCP konekcija počinje sa TCP *three-way handshake*-om, kojem je potreban vremenski period jednog obilaska između klijenta i servera. Ovo je fiksan trošak koji imaju sve http sesije koje nisu trajne odnosno *keep-alive*.

Dolazi se do jednostavne optimizacije: da se ponovo koristi tcp konekcija. Dodavanjem podrške za http trajnu konekciju koja je prikazana na slici 1, omogućeno je to da se eliminiše drugi tcp *three-way handshake* odnosno jedan čitav obilazak sa klijenta na server.



Slika 1. *Keep-alive konekcija*

U slučaju trajne konekcije, vreme koje se uštedi za N zahteva predstavlja (N – 1) * (vreme jednog celog obilaska) u odnosu na primer slanja zahteva bez trajne konekcije.

2.3.2 Korišćenje više TCP konekcija

Zbog nedostatka *multiplexing*-a kod Http-a 1.1, *browser* može naivno da stavi u red sve http zahteve na klijentu i šalje jedan za drugim putem jedne trajne veze. U praksi ovo je vrlo sporo. Zbog toga *browser*-i otvaraju više tcp sesija u paraleli. Ograničenje je da može da se otvori maksimalno šest konekcija za istog *host*-a.

Šest konekcija je odabrano bazirano na broju kompromisa koji se moraju uzeti u obzir. Što je veći broj konekcija to su veći opšti troškovi i na serveru i na klijentu. Međutim sa većim brojem konekcija je i veći broj zahteva koji se mogu obraditi u paraleli. Šest konekcija po *host*-u je siguran kompromis.

2.3.2 Troškovi protokola

Http 0.9 je bio jednostavan, jednolinijski ASCII zahtev za dobavljanje html dokumenata i imao je minimalan trošak. Http 1.0 je proširio ovaj protokol dodajući pojam zaglavlja zahteva i odgovora kako bi omogućio i jednoj i drugoj strani da razmene dodatne informacije o meta-podacima zahteva i odgovora. Konačno, http 1.1 je učinio ovaj format standardom: zaglavlja se lako proširuju i bilo koji server ili klijent ih može proširiti, i takođe se šalju kao običan tekst kako bi ostali kompatabilni sa prethodnim verzijama http-a.

Svaki zahtev koji je bio iniciran od strane *browser*-a će imati dodatnih 500-800 bajtova Http meta-podataka: *user-agent* string, *accept* i *transfer* zaglavlja koja se retko menjaju, *caching directives*, i tako dalje. Međutim u ovih 500-800 bajtova nisu uračunati HTTP *cookie*-i koji mogu najviše resursa zauzimati. Sa pomenutim *cookie*-ima, http meta-podaci mogu zauzimati i po nekoliko kilobajta dodatnih podataka za svaki http zahtev.

Rast liste http zaglavlja nije samo po sebi loša stvar, jer svako zaglavlje postoji s dobrim razlogom. Međutim, činjenica da su sva http zaglavlja običan tekst (bez ikakve kompresije), može dovesti do visokih troškova za svaki zahtev, što može dovesti do kašnjenja i lošijeg korisničkog doživljaja kod nekih aplikacija.

3. HTTP 2.0

Od svog objavljivanja, RFC 2616 je poslužio je kao temelj za neviđeni rast interneta. Milijarde uređaja svih oblika i veličina, od desktop računara do sitnih veb uređaja kao što su telefoni, svakodnevno koriste HTTP za isporuku vesti, video zapisa i miliona drugih veb sadržaja koji se koriste u svakodnevnom životu prosečne osobe.

Ono što je počelo kao jednostavan, jednolinijski protokol za preuzimanje hiperteksta brzo se razvilo u generički transport hipermedija, a sada se, može iskoristiti za isporuku gotovo svakog slučaja korišćenja koji se može zamisliti. Sveprisutnost servera koji mogu da koriste protokol i široka dostupnost klijenata dovele su do toga da su mnoge aplikacije sada dizajnirane isključivo za rad preko HTTP-a.

Jednostavnost HTTP protokola je ono što je omogućilo njegovo prvobitno usvajanje i brz rast. U stvari, sada nije neobično pronaći ugrađene uređaje – senzore, pokretače – koji koriste HTTP kao primaran protokol za kontrolu i podatke. Ali pod težinom sopstvenog uspeha i kako se sve više nastavlja sa migracijom svakodnevnih interakcija na internet – društvene mreže, e-pošta, vesti i video zapisi – takođe je počeo da pokazuje znakove slabosti. Korisnici i veb programeri sada traže reakciju u realnom vremenu i performanse od protokola http 1.1 koje on ne može da isporuči bez nekih izmena samog protokola.

Suočavajući se sa ovim novim izazovima, Http se nastavlja razvijati, i otuda je radna grupa HTTPbis najavila novu inicijativu za Http-om 2.0 [8] početkom 2012. godine. Primarni fokus http-a 2.0 je na poboljšanju transportnih performansi i omogućavanju manjeg kašnjenja i veće propusnosti. Povećanje glavne verzije zvuči kao veliki korak, ali važno je napomenuti da nije pogođena ni jedna semantika protokola na visokom nivou: sva HTTP zaglavlja, vrednosti i slučajevi korišćenja su ostali isti.

Http 2.0 [5] omogučava efikasnije korišćenje mrežnih resursa i smanjenu percepciju kašnjenja tako što uvodi kompresiju polja zaglavlja i omogućava više konkurentnih razmena na istoj konekciji. Tačnije, omogućava preplitanje poruka zahteva i odgovora na istoj konekciji i koristi efikasno kodiranje za http polja zaglavlja.

Takođe, omogućava dodeljivanje prioriteta zahtevima što omogućava da bitniji zahtevi budu pre obrađeni što dodatno poboljšava performanse. Omogućeno je i efikasnije procesiranje poruka kroz korišćenje binarnog *framing*-a poruka.

Važno je za napomenuti da http 2.0 proširuje, ne zamenjuje, prethodne http standarde. Semantike aplikacija koje koriste http su iste i nikakve promene se neće izvesti nad postojećim funkcionalnostima ili suštinskim

pojmovima kao što su http metode, status kodovi ili polja zaglavlja. API na visokom nivou ostaje isti ali su urađene izmene nad niskim nivoom koje utiču na nedostatke performansi prethodnih verzija http-a.
U nastavku će biti obrađeni pojmovi koji nadograđuju prethodni protokol (HTTP 1.1) i poboljšavaju performanse HTTP protokola.

3.1 Binarni *framing* sloj

U srži svih poboljšanja performansi http-a 2.0 jeste novi binarni *framing* sloj koji diktira kako su http poruke enkapsulirane i prenesene između klijenta i servera.
„Sloj" se odnosi na izbor da se predstavi nov, optimizovan mehanizam za šifrovanje između *socket* interfejsa i viših http api-a koji su izloženi aplikaciji. Http semantike kao što su metode, zaglavlja itd. su nepromenjene ali način na koji su šifrovani dok se vrši slanje je drugačije. Nasuprot novom redu unutar običnog teksta http-a 1.1, sva http 2.0 komunikacija je podeljena na manje poruke i *frame*-ove, od kojih je svaki šifrovan (enkodiran) u binarni format.
Kao posledica ovoga, i klijent i server moraju da koriste novi binarni mehanizam šifrovanja da bi se razumeli međusobno. Http 1.1 klijent neće razumeti server koji govori samo http 2.0 i obrnuto. Aplikacije ne moraju znati za ove promene jer klijent i server obavljaju sva neophodna *framing* posla umesto njih.

3.2 *Multiplexing* zahteva i odgovora

U verziji http-a 1.1, da bi klijent napravio više paralelnih zahteva kako bi poboljšao performanse, morao je koristiti više tcp konekcija. Ovakvo ponašanje je direktna posledica mehanizma rada http-a 1.1 pošto on obezbeđuje da se tačno jedan odgovor može dostaviti u toku jednog vremenskog perioda po konekciji. Ovo predstavlja problem ako je prvi odgovor koji treba da se dostavi dugačak pa ostali moraju čekati dok se prvi ne dostavi u celosti. Rezultat je neefikasno korišćenje tcp konekcije.
Binarni *framing* sloj u http-u 2.0 rešava ova ograničenja i omogućava preplitanje odnosno *multiplexing* zahteva i odgovora u celosti tako što dozvoljava klijentu i serveru da razbiju http poruku u posebne *frame*-ove, isprepliću ih i onda ih ponovo sastave na drugom kraju.
Mogućnost da se razbije http poruka u posebne *frame*-ove, isprepliće ih i onda ih ponovo sastavi na drugom kraju je najbitnije unapređenje http-a 2.0.
Novi binarni *framing* sloj u http-u 2.0 rešava problem blokiranja prvog zahteva ili odgovora koji je bio zastupljen u http-u verzije 1.1 i takođe eliminiše potrebu za više tcp konekcija kako bi se omogućio paralelizam pri procesiranju kao i paralelnom dostavljanju zahteva i odgovora. Kao rezultat ovoga, aplikacije su brže, jednostavnije i jeftinije za *deployment*.

3.3 *Server push*

Još jedna moćna novina koju uvodi http 2.0 jeste mogućnost servera da šalje više odgovora za isti klijentski zahtev. Znači, pored odgovora na originalni zahtev, server može da *push*-uje dodatne resurse na klijenta bez da ih je klijent zatražio eksplicitno.
Tipična veb aplikacija se sastoji od mnogo resursa, koji se otkrivaju od strane klijenta tako što klijent isčitava dokumente koji su poslati sa servera i kada naiđe u tom dokumentu na neku vezu sa drugim dokumentom ili resursom, onda ga zatraži od servera. Ovo zahteva, dodatno vreme koje bi se moglo smanjiti tako što će server sam da pošalje dodatne dokumente ili resurse koje klijent zahteva za rad sa inicijalnim dokumentom i time znatno smanjiti vreme čekanja klijenta.

3.4 Kompresija zaglavlja

Svaki http zahtev ili odgovor nosi sa sobom i zaglavlje koje opisuje resurse koji se prenose kao i njihova obeležja. U http-u 1.1, ovi meta-podaci se uvek šalju kao običan tekst što dodaje na zahtev oko 500 do 800 bajtova dodatnih podataka pri prenosu i nekada po više kilobajta ako postoje http *cookie*-i koji se koriste. Kako bi se smanjila ova količina podataka i samim tim i poboljšale performanse, http 2.0 vrši kompresiju zaglavlja http zahteva i odgovora koristeći HPACK kompresioni format koji koristi dve jednostavne ali i moćne tehnike:

- dozvoljava poljima zaglavlja koja se šalju da budu komprimovana omoću Huffman-ovog koda koji smanjuje veličinu svakog zahteva i odgovora koji se šalje,
- zahteva da i klijent i server održavaju i ažuriraju indeksiranu listu prethodno viđenih polja zaglavlja koja se koristi kao referenca za efikasno komprimovanje prethodno prenošenih vrednosti.

Huffman-ovo kodiranje dovoljava individualnim vrednostima da budu komprimovane kada se šalju, dok indeksirana lista prethodno prenošenih vrednosti dozvoljava da se komprimuju duplicirane vrednosti tako što se prenose indeksi vrednosti koji se mogu koristiti da efikasno pretraže i rekonstruišu ključeve i vrednosti zaglavlja.

4. PODEŠAVANJE HTTP-A 2.0 NA POPULARNIM SERVERSKIM TEHNOLOGIJAMA

4.1 Podešavanje HTTP-a 2.0 u *spring boot* aplikaciji

Podrška za http 2.0 u *spring boot* aplikaciji se može omogućiti sa konfiguracionim obeležjem *server.http2.enabled*. Ova podrška zavisi od odabranog servera i aplikacijskog okruženja pošto protokol nije podržan po *default*-u u JDK-u 8.

4.2 Podešavanje HTTP-a 2.0 u *node.js* aplikaciji

Core API pruža *low-level* interfejs koji je dizajniran da podrži karakterstike http 2.0 protokola. Nije dizajniran za kompatibilnost sa postojećim http 1 modul api-em, ali Compatibility API jeste.

4.3 Podešavanje HTTP-a 2.0 u ASP.NET Core aplikaciji

Veb server koji se koristi po *default*-u u asp.net Core projektu se naziva Kestrel. HTTP 2.0 je podržan u asp.net Core aplikacijama koje koriste Kestrel ako su zadovoljeni sledeći uslovi:

- Koristi se operativni sistem:
  - Windows Server 2016/Windows 10 ili noviji ili
  - Linux sa OpenSSL-om 1.0.2 ili noviji (npr. Ubuntu 16.04 ili noviji).
- Ciljani framework: .net Core 2.2 ili noviji,
- Application-Layer Protocol Negotiation (ALPN) konekcija, i
- TLS 1.2 ili novija konekcija.

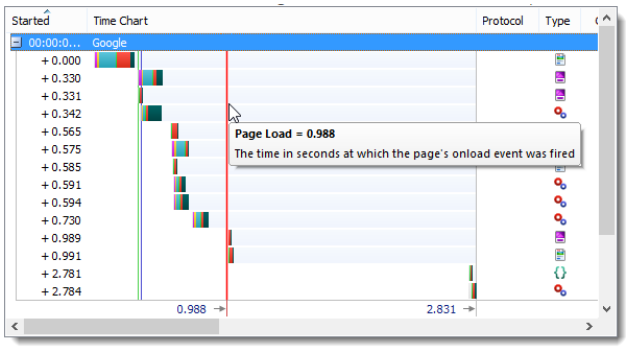
Ako su prethodno navedeni uslovi zadovoljeni HTTP 2.0 je omogućen po *default*-u i nisu potrebne nikakve dodatne konfiguracije.

5. PERFORMANSE HTTP-A 2.0

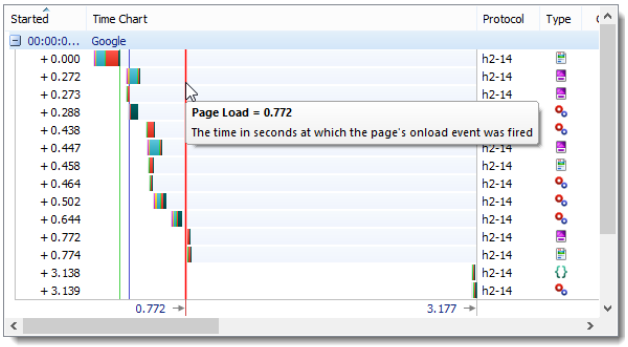
Za ocenu performansi HTTP-a 2.0, poredio se ovaj protokol sa njegovom prethodnom verzijom.

5.1 Vreme učitavanja stranice

Događaj učitavanja u *HttpWatch*-u pokazuje kada se stranica skinula u potpunosti i kada je spremna za korišćenje. U većini slučajeva ovo je razumna mera brzine stranice iz perspektive posetioca veb sajta.
Na slikama 2 i 3 su prikazane brzine učitavanja stranica za http 1.1 i 2.0.



Slika 2. *Brzina učitavanja stranice kod HTTP-a 1.1 [6]*



Slika 3. *Brzina učitavanja stranice kod HTTP-a 2.0 [6]*

Sa slika 2 i 3 se vidi da je brzina učitavanja stranice kod http-a 1.1 sporija za 0.216 sekundi u odnosu na HTTP 2.0 što je 21.86% (100 – (0.772 / 0.988 * 100)). Ovakva razlika u performansama je izražena zbog nekomprimovanih zaglavlja i dodatnih tcp konekcija i ssl *handshake*-ova kod protokola 1.1. Za kompleksnije stranice, brzina http-a 2.0 bi trebala da bude još izraženija.

6. ZAKLJUČAK

Na osnovu teorijske obrade protokola http 1.1 i http 2.0, može se ustanoviti da je http 2.0 bolji protokol u svakom pogledu jer samo popunjava nedostatke prošlog protokola i dodaje neke nove funkcionalnosti dok stare ostaju iste. Dodatna prednost ovog protokola jeste da unapređivanje protokola sa verzije 1.1 na 2.0 neće zahtevati promene u aplikaciji sem onih na visokom nivou kao što su konfiguracije.

Na osnovu poređenja performansi ova dva protokola se takođe može utvrditi da je http 2.0 superiorniji protokol. U testovima je pokazano da je vreme učitavanja stranice smanjeno za 21.86% sa upotrebom HTTP-a 2.0. Ovo su znatna poboljšanja performansi protokola, a za kompleksnije stranice, bi mogle doći još više do izražaja prednosti HTTP-a 2.0 što se tiče smanjenja vremena učitavanja stranica.
Podešavanje HTTP-a 2.0 za popularne *backend* tehnologije je relativno jednostavno ukoliko se koristi https dok je za nešifrovan http saobraćaj uglavnom nemoguće podesiti http 2.0. Ovo predstavlja veliku manu protokola, jer mnogi veb sajtovi ne zahtevaju šifrovan saobraćaj, a zbog toga ne mogu da koriste prednosti http-a verzije 2.0.
Iako je pokazano da se i *server push* može lako implementirati u popularnim *backend* tehnologijama, za razliku od ostalih poboljšanja protokola koje su pomenute, performanse koje se mogu dobiti (ili izgubiti) pomoću ove funkcionalnosti zavise od programera, odnosno od implementacije *endpoint*-a i odabira koji resursi će biti *push*-ovani.

7. LITERATURA

[1] HTTP: https://www.w3.org/History/19921103-hypertext/hypertext/WWW/Protocols/HTTP.html
[2] World Wide Web: https://en.wikipedia.org/wiki/World_Wide_Web
[3] HTTP 0.9: https://hpbn.co/brief-history-of-http/#http-09-the-one-line-protocol
[4] HTTP 1.0: https://hpbn.co/brief-history-of-http/#http10-rapid-growth-and-informational-rfc, https://hpbn.co/http1x/
[5] HTTP 2.0: https://tools.ietf.org/html/rfc7540, https://hpbn.co/brief-history-of-http/#http2-improving-transport-performance, https://hpbn.co/http2/
[6] Poređenje performansi HTTP-a 1.1 i 2.0: https://blog.httpwatch.com/2015/01/16/a-simple-performance-comparison-of-https-spdy-and-http2/

Kratka biografija:

Marko Krajinović, rođen 19.11.1995 u Somboru. Završio je osnovnu školu Nikola Vukićević u Somboru i srednju tehničku školu u Somboru. Diplomirao je na fakultetu tehničkih nauka u Novom Sadu.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.38
DOI: https://doi.org/10.24867/07BE30Bozic

PRIMENA PID REGULATORA U STABILIZACIJI I UPRAVLJANJU KRETANJEM KVADKOPTERA

APPLICATION OF PID REGULATORS IN STABILIZATION AND MOTION CONTROL OF QUADCOPTER

Milan Božić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je prikazan jedan od načina implementacije letelice na daljinsko upravljanje – kvadkoptera za čije upravljanje se koristi STM32F103C8T6 kontroler, za određivanje položaja letelice koristi se inercijalni senzor IMU MPU9250 (akcelerometar i žiroskop) dok se za aktuatore koriste motori bez četkica. Senzorski i aktuatorski deo povezuje tri PID regulatora koji su implementirani u kontroleru. Upravljanje se vrši radio vezom na 2.4GHz modulom za komunikaciju NRF24l01. Letelica obezbeđuje podatke o temperaturi, vlažnosti i atmosferskom pritisku uz pomoć senzora BME280.*

Ključne reči: *dron, kvadkopter, akcelerometar, žiroskop, PID, BME280, NRF24L01, MPU6050, MPU9250, STM32*

Abstract – *The paper presents one of the ways of implementing a remote control aircraft - a quadcopter whose STM32F103C8T6 controller is used, to determine the position of the aircraft using an inertial IMU MPU9250 sensor (accelerometer and gyroscope) while brushless motors are used for actuators. The sensor and actuator part connects the three PID controllers implemented in the controller. It is controlled by radio communication on the 2.4GHz communication module NRF24l01. The aircraft provides temperature, humidity and atmospheric pressure data using a BME280 sensor.*

Keywords: *drone, quadcopter, accelerometer, gyroscope, PID, BME280, NRF24L01, MPU6050, MPU9250, STM32*

1. UVOD

Veoma često je u praksi za rešavanje različitih zadataka potrebno da letelice imaju mogućnost vertikalnog poletanja i sletanja. Dronovi su u ovoj oblasti nenadmašni. Kvadkopteri predstavljaju podvrstu letelica na daljnsko upravljanje koji se odlikuju osobinom da sadrže četri uzgonska motora čijim se upravljanjem vrši kretanje letelice. Danas se kvadkopteri nalaze u različitim aspektima života, od vojne industrije do industrije zabave. Različiti zadaci zahtevaju različite performanse. Na primer, u vojnoj industriji potrebno je da letelica bude pouzdana i robustna i da predstavlja sistem otporan na otkaze, dok je u industriji zabave potrebno da letelica ima što bolje dinamičke osobine - da može da napravi različite akrobatske pokrete i da bude što atraktivnija i zabavnija. I jednu i drugu vrstu letelica povezuju zajednički zahtevi za senzorima i aktuatorima, s tom razlikom što je negde potrebno koristiti senzor veće pouzdanosti i rezolucije merenja, negde je potrebno koristiti motore sa jačim potiskom.

Ovaj rad će proći kroz sve potrebne segmente za izradu ove vrste letelice dajući opšte šeme kao i postulate koje je potrebno koristiti kako bi se postigli što bolji rezultati leta. Akcenat se stavlja na implementaciju *PID* (eng. *proportional–integral–derivative*) algoritma za stabilizaciju i kretanje kvadkoptera. Izrađeni kvadkopter predstavlja niskobudžetnu letelicu sa radijusom kretanja do 1km. Letelica nema implementiran algoritam za stabilizaciju visine (eng. *auto leveling*).

2. ALGORITAM SENZORA *MPU9250* (*MPU6050*)

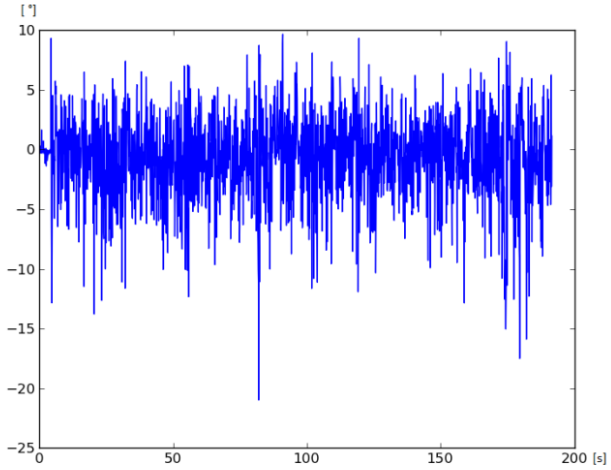
Za upravljanje letelicom pre svega potrebno je dobro izvršiti čitanje trenutnog položaja. Ovo se vrši kombinacijom senzora akcelerometra i žiroskopa. U ovom radu korišćen je devetoosni senzorski sistem sastavljen od troosnog akcelerometra, troosnog žiroskopa i troosnog magnetometra. U projektu nije bilo potrebe za korišćenjem magnetometra. Korišćeni senzor *MPU9250* sastavljen je od dva senzora *MPU6050* i *AK8963*, akcelerometar + žiroskop i magnetometar, redom. Kako magnetometar nije korišćen algoritam koji će biti predstavljen potpuno će odgovarati i senzoru *MPU6050* kao i *MPU6500* koji predstavlja unapređenu generaciju *IMU* senzora.

Apsolutni položaj letelice bi se mogao odrediti samo uz pomoć akcelerometra kada bi se vibracije koje proizvode motori i propeleri svele na zanemarljive. Kako je akcelerometar veoma osetljiv na vibracije njegov izlaz je potrebno filtrirati niskopropusnim filtrom i tada njegov signal zbog kašnjenja koje filtar unosi postaje neupotrebljiv za upravljanje letelicom u realnom vremenu.

Na slici 1 dat je prikaz izlaza akcelerometra pri vibracijama proizvedenim od strane rotora. Takodje vidimo da je dugoročno gledano signal konzistentan, ali kao što je već rečeno potrebno ga je filtrirati i nije ga moguće koristiti za kontrolisanje letelice u realnom vremenu.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Vladimir Rajs, docent.

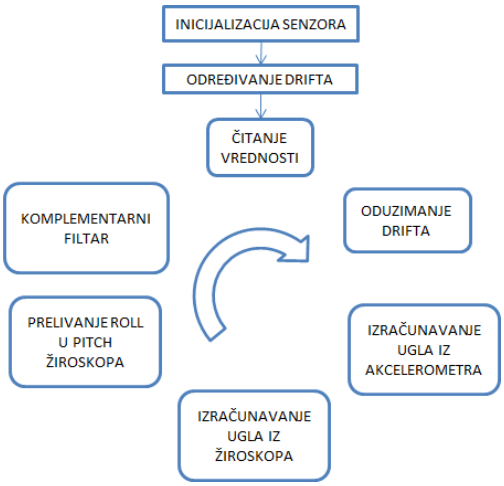
Bitno je naglasiti da signal akcelerometra nema promene vrednosti u stanju mirovanja (eng. *drift*) tokom vremena i iz tog razloga je veoma bitan za ove vrste letelica.



Slika 1. *Odziv akcelerometra uz vibracije rotora* [1]

Žiroskop kao senzor koji ima mogućnost da detektuje brzinu kretanja ovde ga možemo iskorisititi da bismo prikupili sva kretanja i što preciznije odredili trenutni položaj senzora, a samim tim i letelice. Problem koji se neminovno javlja kod žiroskopa je šum koji se sjedinjava sa signalom. Kombinacija akcelerometra i žiroskopa daju senzor koji je moguće korisititi za stabilizaciju letelice.

Na slici 2 data je blok šema algoritma koji je potrebno implementirati za pravilno korišćenje žiroskopa i akcelerometra.



Slika 2. *Algoritam za odredjivanje ugla iz ziroskopa i akcelerometra*

Pod *iniclijalizacijom senzora* podrazumeva se: podešavanje njegove rezolucije, mogućnost određivanja semplovanja kao i testiranja ispravnosti senzora. Oduzimanje *drifta* žiroskopa podrazumeva sabiranje određenog broja odbiraka kada senzor miruje zatim deljenje sa brojem odbiraka. Ova dva koraka se izvršavaju pri uključivanju senzora i nakon toga se ciklično smenjuju ostali koraci.

Prvi u seriji koraka je da se pročitaju vrednosti iz senzora, zatim je potrebno oduzeti predhodno određenu vrednost za neutralisanje šuma što predstavlja naredni korak. Kada su podaci pročitani moguće je odrediti uglove iz akcelerometra i žiroskopa što su naredna dva koraka. Naredni korak ima zadatak da pri rotaciji letelice oko Z ose vrši prelivanje *roll*-a u *pitch* i obrnuto u zavisnosti od sinusa ugla rotacije. Nakon što su uglovi pravilno izračunati da bi se dugotrajno osiguralo merenje bez šuma dodaje se komplementarni filtar koji uzima 1% vrednosti izračunatih u akcelerometru i 99% vrednosti iz žiroskopa. Ovaj ciklus čitanja potrebno je vršiti što češće kako bi merenje bilo što preciznije, u praksi je dovoljno izračunavati brže od 250Hz [2]. U realizovanom sistemu, ciklus se odvija brzinom od 500Hz.

3. *PID* REGULATOR

Za stabilizaciju letelice korišćen je *PID* kontroler po tri ose rotacije. Dva identična *PID*-a po *roll* i *pitch* i još jedan kontroler po *yaw [3]*.

U nastavu je dat korišćeni *PID* regulator.

P = koefP * error

I += koefI * error

D =koefD*(delta_error[0]+delta_error[1]+ delta_error[2])

PID = P+I+D

- *koefP , koefI i koefD* predstavljaju koeficijente regulatora koje je potrebno podesiti.
- *error* predstavlja trenutnu grešku,
- *delta_error* predstavlja poslednje tri razlike *error*-a. Možemo reći da daje informaciju o napredovanju greške.

D regulator razlikuje se od klasičnog D regulatora iz razloga što prikuplja informacije i o prethodnim delta *error*-ima. Prednost ovog pristupa je što ako se javi lažna greška zbog vibracija akcelerometra ona će u velikoj meri biti potisnuta od sledeće dve delta greške. Ovaj pristup smanjuje vibracije koje se dobijaju od nesavršenosti senzora.

Koeficijenti koji daju najbolje rezultate su :

koefP = 0.52 , koefD = 15.5 , koefI = 0.001

Dati koeficijenti daju najbolje rezultate za izrađeni model i nece imati najbolje rezultate na nekom dugom hardveru ali se mogu koristiti kao startni koeficijenti sa podešavanje stabilizacije.

U izrađenoj letelici koristi se laserski senzor udaljenosti *ToF (*eng. *TimeOfFlight) VL53L01* kako bi se odredila udaljenost od podloge. Ako se detektuje da se letelica nalazi na zemlji tada se iskuljučuje I regulator - ovim se povećava brzina stabilizacije letelice jer se sprečava nekontrolisan rast I regulatora dok se letelica ne odvoji od zemlje.

4. KOMUNIKACIJA

Komunikacija između upravljačke jedinice (eng. *joystick*) i letelice ostvarena je uz pomoć komunikacijonog modula *NRF24L01*. Ovaj primo-predajnik obezbeđuje razmenu informacija između leteice i upravljača u oba smera, za upravljanje letom upravljački signali šalju se od upravljača ka letelici dok se u suprotnom smeru šalju informacije o bateriji, trenutnoj visini , pritisku vazduha , temperaturi i vlažnosti vazduha.

Poruke koje se razmenjuju su veličine 8 bajta. U nastavku su date tabele podataka koji se razmenjuju.

Tabela 1. *Raspored podataka* (upr. jed. - letelica)

0	1	2	3	4	5	6	7
UD	LR	FB	Rot	Cam	CamON	non	non

UD – Opšta brzina svih motora
LR – Skretanje po Roll-u
FB – Skretanje po Pitch-u
Rot – Rotacija
Cam – Pomeranje kamere
CamON – Uključivanje kamere
non – Nekoristi se

Tabela 2. *Raspored podataka* (letelica – upr. jed.)

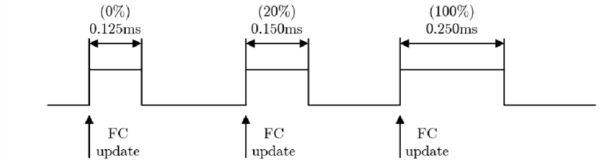
0	1	2	3	4	5	6	7
Vbat		Ibat	Hum	Temp	Press	Alt	non

Vbat – Napon baterije
Ibat – Struja baterije
Hum – Vlažnost vazduha
Temp – Temperatura
Press – Pritisak
Alt - Visina

Brzina prenosa podataka je na minimalnoj brzini od 250kbps čime se postiže maksimalan domet radio veze od 1.1km. Potrebno je naglasiti i da se kanal prenosa nalazi na 108 od 125 kako bi se nosač našao iznad frekvencije korišćenog nosača za bežičnu komunikaciju (eng. *wire-less*) mreže kako se bi se izbegla interferencija sa drugim signalima u istom frekvencijskom spektru u toku prenosa.

5. MOTORI

Koriste se četri motora bez četkica (eng. *brushless*) i na njima se nalaze propeleri *LjI 5045*. Motori su pogonjeni *ESC* (eng. *electronic speed control*) *EMAX BLHeli 30A*. Komunikacioni protokol za upravljanje *ESC*-ovima koji se koristi je *OneShot 125*. Ovo predstavlja osam puta brži protokol od klasičnog *PWM* (eng. *Pulse Width Modulation*) protokola. Neophodnost korišćenja ovog prokokola je iz razloga što je potrebno da *PID* petlja a samim tim i upravljanje motorima bude veće od 250Hz ovo nije moguće da klasičnim *PWM*-om koji radi na 50Hz. Sa *OneShot125* protokolom moguće je vršiti kontrolu motora i do osam puta brže. Specifičnost ovog protokola je da se širina impulsa menja od 125-250µS dok je perioda 2.5mS. Na Slici 3 prikazan je signal datog protokola.



Slika 3. *Protokol OneShot 125* [4]

Minimalna razlika koju ESC može da detektuje između trajanja impulsa je 1µs tako da se zapravo postoji 125 različitih upravljačkih signala.

6. HMI EKRAN

Za prikaz informacija na upravljačkoj jedinici koristi se *HMI* (eng. *Human-Machine Interface*) Ekran (eng. *display)*. Ovo predstavlja vrlo jednostavan način za prikaz informacija. *HMI* sastoji se od ekrana osetljivog na dodir i kontrolera. Za programiranje interfejsa koristi se *Nextion editor*. Upravljanje ekranom vrši se asinhronom *UART* komunikacijom sa brzinom od 9600.
Prednost ovakve vrste prikaza je što već unapred postoje predefinisane funkcije za iscrtavanje grafika , dodavanje dugmadi, tekstova itd. Korišćeni ekran je veličine 2.8".

7. SENZOR BME280

Za određivanje osnovnih parametara vazduha u kom se nalazi letelica koristi se senzor *BME 280*. Ovaj senzor ima mogućnost merenja temperature, pritiska i vlažnosti vazduha. Ima vrlo oseljiv senzor pritiska pa je moguće odrediti nadmorsku visinu u odnosu na pritisak.

Matematička jednačina koja povezuje pritisak sa nadmorskom visinom data je u nastavku :

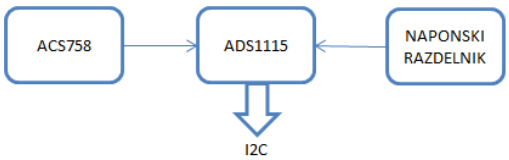
$$visina = 44330 \cdot \left(1 - \left(\frac{pritisak}{pritisak\_nivoa\_mora}\right)^{0.1903}\right)$$

Jednačina 1. *Određivanje nadmorske visine* [5]

Najveća nadmorska visina ostvarena pri letu ostaje zabeležena i može se očitati na ekranu upravljačke jedinice. Kao i ostali podaci obezbeđeni od strane ovog senzora.

8. MERENJE STRUJE I NAPONA

Merenje struje koja se povlači iz baterije meri se uz pomoć *ACS758*. Ovo je senzor koji meri magnetno polje koje proizvodi kretanje struje kroz provodnik. Senzor je baziran na *hall*-ovom efektu i odlikuje ga veoma niska otpornost okvirno oko 100uΩ. Izlaz ovog senzora dovodi se na ulaz *ADS1115* integrisanog četorokanalnog *AD* konvertora koji je podešen da meri napon u opsegu +-4.096V. *ADS1115* je 16-bitni *AD* konvertor koji se koristi i za precizno merenje napona baterije. Kako se u projektu koristi 3S (tri serijski vezane ćelije) li-po baterija maksimalnog izlaznog napona 12.6V, merenje je potrebno izvršiti preko naponskog razdelnika koji je sastavljen od otpornika tolerancije 1% otpornosti 1k5Ω i 4k7Ω. Podaci o struji i naponu prikazuju se na ekranu upravljačke jedinice. Na slici 4 data je blok šema gore navedenog mernog sistema.



Slika 4. *Blok šema mernog sistema na letelici*

9. REZULTATI

Testiranja pokazuju da je dobijena funkcionalna letelica. Odmah nakon poletanja u vremenu od 200ms u mogućnosti je da uspostavi stabilnost. Za ovo je zaslužan ranije pomenut sistem za isključivanje iregulatora.

S obzirom na to da letelica nema regulator koji bi držao visinu fiksnom (eng. *auto leveling*) može se reći da je letelica prilično upravljiva i različite visine moguće je održavati menjanjem snaga motora.

Odziv letelice koji je direktno zavisan od korišćenog komunikacionog modula radi veoma stabilno bez gubljenja poruka kada je između prijemnika i predajnika omogućena optička vidljivost i kada je njihovo rastojanje do 200m. Korišćenjem snažnijeg primo-predajnika sa većim dometom koji bi brže slao poruke postigao bi se upravljiviji sistem.
Merenje parametara vazduha temperature, vlaznosti i pritiska koji se dobijaju iz *BME 280* senzora potrebno je testirati kako bi se potvrdili dobijeni rezultati i detektovala moguća odstupanja od tačnih vrednosti.

10. ZAKLJUČAK

U ovom radu predstavljen je jedan od načina implementacije letelice sa četiri motora sa vertikalnim potiskom, kvadkoptera. Implementirano rešenje zasniva se na algoritmu korišćenom u letelicama korišćenim za zabavu koji koriste veoma jeftine komponente (*MPU6050*, *NRF24l01*). Korišćenjem kvalitetnijih senzora položaja kvalitet upravljanja bi se zanatno poboljšao. Takođe, treba istaći da je ceo projekat zasnovan na *STM32F103C8T6* jeftinom mikrokontroleru sa vrlo ograničenim mogućnostima.

Korišćenjem bržeg kontrolera omogućili bi povećanje brzine čitanja sa senzora položaja kao i povećanje brzine izračunavanja PID petlje. Ograničavajući faktor u poboljšanju letelice jeste i sam *OneShot125* protokol koji ograničava upravljanje na 400Hz.

Za veće brzine potrebno je korisiti neki napredniji upravljački protokol kao što su *OneShot42*, *Dshot*, *MultiShot* koji predstavljaju mnogo brže protokole [6]. Povećanje brzine i kvaliteta letenja podrazumeva i brisanje epiteta niskobudžetne letelice iz gore navedenih razloga.

11. LITERATURA

[1] https://i.stack.imgur.com/4fA1L.png (pristupljeno u oktobru 2019.)
[2] http://www.brokking.net/ymfc-al_main.html (pristupljeno u oktobru 2019.)
[3] M. F. Silva *et al*., "Design of angular PID controllers for quadcopters built with low cost equipment," *2016 20th International Conference on System Theory, Control and Computing (ICSTCC)*, Sinaia, 2016
[4] https://www.researchgate.net/figure/An-example-of-the-PWM-signal-under-the-OneShot125-protocol_fig12_327417180 (pristupljeno u oktobru 2019.)
[5] W. Zhu, Y. Dong, G. Wang, Z. Qiao and F. Gao, "High-precision barometric altitude measurement method and technology," *2013 IEEE International Conference on Information and Automation (ICIA)*, Yinchuan, 2013
[6] https://quadmeup.com/pwm-oneshot125-oneshot42-and-multishot-comparison/ (pristupljeno u oktobru 2019.)

Kratka biografija:

Milan Božić rođen je u Rumi 1995. god. Osnovnu školu "Dositej Obradović" završio je 2010.godine u Putincima. Tehničku školu "Milenko Brzak - Uča" u Rumi završio je 2014.godine. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Primenjena elektronika odbranio je 2018.god.
kontakt: milan.bozic@uns.ac.rs

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.41
DOI: https://doi.org/10.24867/07BE33Ninkovic

RAZVOJ APLIKACIJE ZA PRAĆENJE RADA PROCESA

DEVELOPMENT OF A PROCESS MONITORING APPLICATION

Isidora Ninković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIČKO I RAČUNARSKO INŽENJERSTVO

Kratak sadržaj – *Ovaj rad se bavi softverskim alatom koji služi za nadgledanje i procenu rada izabranih procesa. Nadgledanje rada procesa predstavlja redovno praćenje i evidentiranje njihovih aktivnosti. Potreba za nadgledanjem rada izabranih procesa proističe iz potrebe za informisanjem o performansama rada programa kojem procesi pripadaju.*

Ključne reči: *nadgledanje procesa, performanse, evaluacija rada, izveštaj o radu*

Abstract – *Software tool for process monitoring and evaluating activities of selected processes is presented in this paper. Monitoring is the regular observation of the processes and recording of their activities. The requisit for activity observation of selected processes arises from the need for perfrormance information about processes which belongs to programme of interest.*

Keywords: *process monitoring, performances, evaluation of work, report about work*

1. UVOD

Kvalitet proizvoda ne bi trebalo da bude kompromis između troškova i karakteristika proizvoda, već ga je nužno ostvariti kako bi proizvod bio što primenljiviji, dugotrajniji, mogao da radi što pouzdanije sa dobrim performansama, ali i kako bi njegovi korisnici bili što zadovoljniji. Ukoliko je proizvod kvalitetan na dugoročnom planu se smanjuju troškovi održavanja i stiču se lojalni korisnici [1].

Kada je reč o kvalitetu softverskog proizvoda podrazumeva se, pre svega, rad uz dobre performanse i pružanje pouzdane usluge, ali i lakoća i intuitivnost korišćenja za širi spektar korisnika, robusnost, određen nivo sigurnosti i bezbednosti. Kako bi se izgradio softverski proizvod koji karakteriše kvalitet na visokom nivou neophodno je pratiti njegov rad još od prvih stadijuma razvoja.

Nadgledanje rada procesa koji pripadaju programu od interesa obuhvata svakodnevno rutinsko prikupljanje podataka o svim aspektima njihovog rada. Prikupljene informacije su značajne na različitim poljima. Tokom faza razvoja softverskog proizvoda softver arhitekte moraju prikupiti što više informacija o radu kako bi doneli ispravne odluke o arhitekturi aplikacije, mogućim izmenama koda, ali i o tome da li se aplikacija ponaša kako je predviđeno. Nadalje su prikupljene informacije od interesa menadžerima, koji na osnovu njih vrše procenu da li je proizvod u skladu sa trenutnim standardima i da li je dovoljno kompetativan na tržištu, kao i neophodnosti izmene njegove cene.

Rad procesa se može pratiti i preko grafikona, na kojima je predstavljeno kretanje određene vrednosti u vremenu. Grafikoni mogu biti u realnom vremenu ili kreirani za zadati vremenski interval, a preko njih se svakako može otkriti neobično ponašanje procesa programa [2].

Nakon prikupljanja podataka o radu može se napraviti i statistika rada procesa, koja se prikazuje u vidu izveštaja. Putem izveštaja korisnik dobija uvid u istorijske podatke, ali i izračunate podatke od interesa, i može izvršiti poređenje sa podacima iz prethodnih izveštaja.

2. SOFTVERSKI ALAT ZA NADGLEDANJE RADA PROCESA

Softverski alat opisan u ovom radu prevashodno služi za nadgledanje aktivnosti procesa. Aktivnosti procesa se ogledaju kroz njihove osnovne karakteristike rada. Mogu se i grafički predstaviti putem grafikona, kako bi se dobio slikovit prikaz kretanja izabrane karakteristike rada za određen vremenski period.

Posebna funkcionalnost ovog softverskog alata jeste mogućnost prilagođenja tako da se ostvari povezivanje sa drugim softverskim proizvodom koji korisnik želi da nadgleda. Tačnije, može se izvršiti nadgledanje izabrane aktivnosti željenog softverskog proizvoda.

2.1 Skup procesa koji se prate

Softverski alat koji je predmet ovog rada karakteriše velika mogućnost prilagođenja korisniku, počev od definisanja šireg skupa procesa koji se prate. Skup procesa od interesa se nalazi u xml konfiguracionom fajlu, i korisnik ga može ručno izmeniti putem navođenja procesa koje želi da prati. Procesi se u xml konfiguracionom fajlu navode putem svog naziva, kao što je prikazano na slici 1. Xml konfiguracioni fajl sa skupom procesa koji se žele nadgledati se učitava prilikom pokretanja aplikacije, tako da je sve izmene neophodno pre pokretanja uneti.

Izgled početnog prozora koji se korisniku otvara prilikom pokretanja softverskog alata je prikazan na slici 2. U tabu *Processes* korisnik dobija pregled procesa iz unapred definisanog skupa, a svaki proces je naveden po nazivu i jedinstvenom identifikatoru (Id). Nakon naziva i Id-ja procesa su navedene osnovne karakteristika njegovog rada:

NAPOMENA:

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

1. *CPU* – trenutno zauzeće procesora od strane procesa, izraženo u procentima,
2. *Memory* – trenutno zauzeće memorije računara od strane procesa, izraženo u MB, i
3. *RAM* – trenutno zauzeće *RAM* (*Random Access Memory*) memorije računara od strane procesa, izraženo u MB.

Karakteristike rada procesa se osvežavaju na svakih 1 sekund, tako da korisnik ima uvid u najsvežije podatke.

all_processes.xml

```
<?xml version="1.0"?>
<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSche
  <string>StabilityAnalysesService</string>
  <string>JobScheduler</string>
  <string>CalculationExecutor</string>
  <string>CoreFunctionsService</string>
  <string>PowerControlSystemService</string>
  <string>FunctionHost</string>
  <string>OutrageProcessingService</string>
  <string>XACTCommandQSpooler</string>
  <string>LocalContextManager</string>
  <string>LoadForecastService</string>
  <string>ModelManagerService</string>
  <string>ModelTopologyService</string>
```

Slika 1. *Xml fajl za konfiguraciju liste procesa od interesa*

Process Monitor

Processes | Performance charts

Filter processes | Monitoring

Name	Id	CPU [%]	Memory [MB]	RAM [MB]
DMS_StabilityAnalysesService	476	0	555.7	365.9
DMS_JobScheduler	15776	0	124.8	91.6
DMS_CalculationExecutor	25380	0	66.1	45.1
DMS_CalculationExecutor	7044	0	66.5	45
DMS_CalculationExecutor	22096	0	66.1	45
DMS_CalculationExecutor	25380	0	66.1	45.1
DMS_CalculationExecutor	7044	0	66.5	45
DMS_CalculationExecutor	22096	0	66.1	45
DMS_CalculationExecutor	25380	0	66.1	45.1

Slika 2. *Izgled početnog prozora*

2.2 Istorijska baza podataka

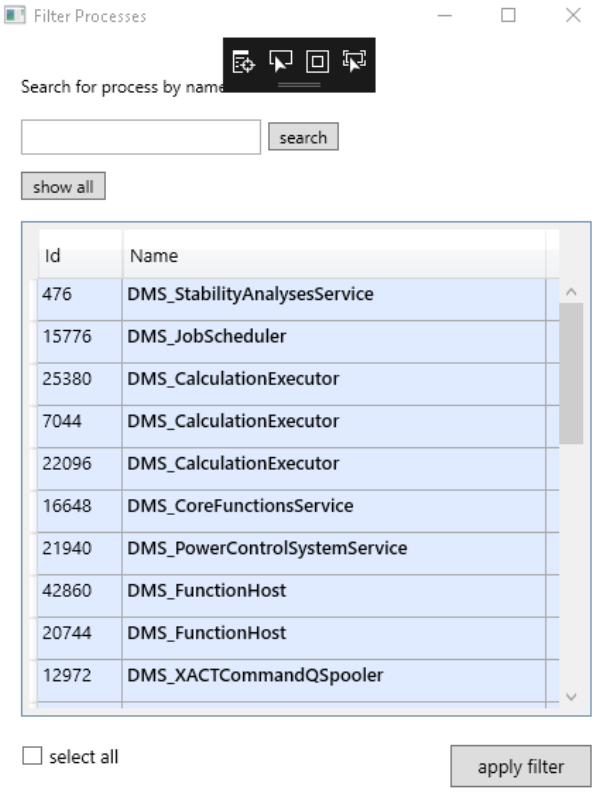
Prilikom očitavanja novih vrednosti karakteristika rada na svakih 1 sekund vrši se njihov prikaz za korisnika, ali i prethodno zapisivanje u istorijsku bazu podataka.

Za potrebe ovog softverskog alata je korišćena *SQLite* baza podataka. *SQLite* je mala, brza baza podataka koja se lako može ugraditi u aplikaciju. Posebna je po tome što čuva sve podatke u samo jednom fajlu. Ne koristi spoljne biblioteke i interfejse, osim nekoliko standardnih C biblioteka, što takođe predstavlja njenu prednost [3].

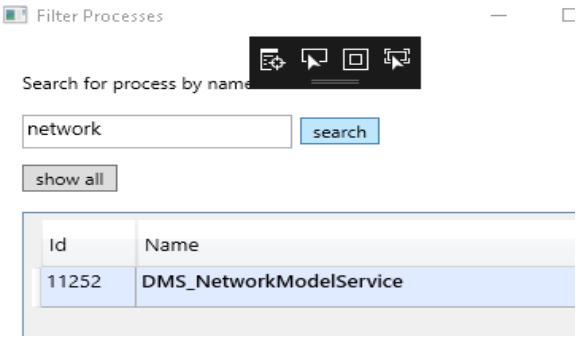
Potreba za istorijskom bazom podataka postoji zbog opcije kreiranja grafičkog prikaza koja se nudi korisniku u okviru taba *Performance charts*, što će naknadno biti opisano u poglavlju 2.3. Takođe, u sastavu posebne funkcionalnosti ovog softverskog alata se nakon praćenja određene aktivnosti procesa kreira izveštaj. Kako bi se kreirao izveštaj za vremenski period trajanja aktivnosti procesa neophodni su istorijski podaci o njegovom radu. Posebna funkcionalost softverskog alata koji je predmet ovog rada će detaljnije biti opisana u poglavlju 2.4.

2.3 Filtriranje procesa

U okviru taba *Processes* korisniku se nudi i opcija dodatnog filtriranja skupa procesa za praćenje, putem klika na dugme *Filter processes*. Po izboru ove opcije pojavljuje se dodatan prozor sa listom svih dostupnih procesa za praćenje i mogućnošću selekcije njenog podskupa. Svi dostupni procesi za praćenje jesu procesi koje je korisnik naveo u xml konfiguracionom fajlu pre pokretanja aplikacije. Korisnik može pretražiti listu po nazivu i Id-ju procesa ukoliko je to neophodno zbog dužine liste i smanjene mogućnosti snalaženja. Nakon izvršene selekcije i klika na dugme *Apply filter* se filter primenjuje, i samo izabrani procesi nastavljaju da se prate. Nakon gašenja aplikacije filtrirana lista se čuva, tako da kada korisnik pokrene aplikaciju sledeći put i dalje će imati samo prikaz filtirane liste procesa. Izgled prozora za filtriranje je prikazan na slici 3, dok primer pretrage je prikazan na slici 4.



Slika 3. *Izgled prozora za filtriranje*



Slika 4. *Primer pretrage*

2.4 Grafici performansi

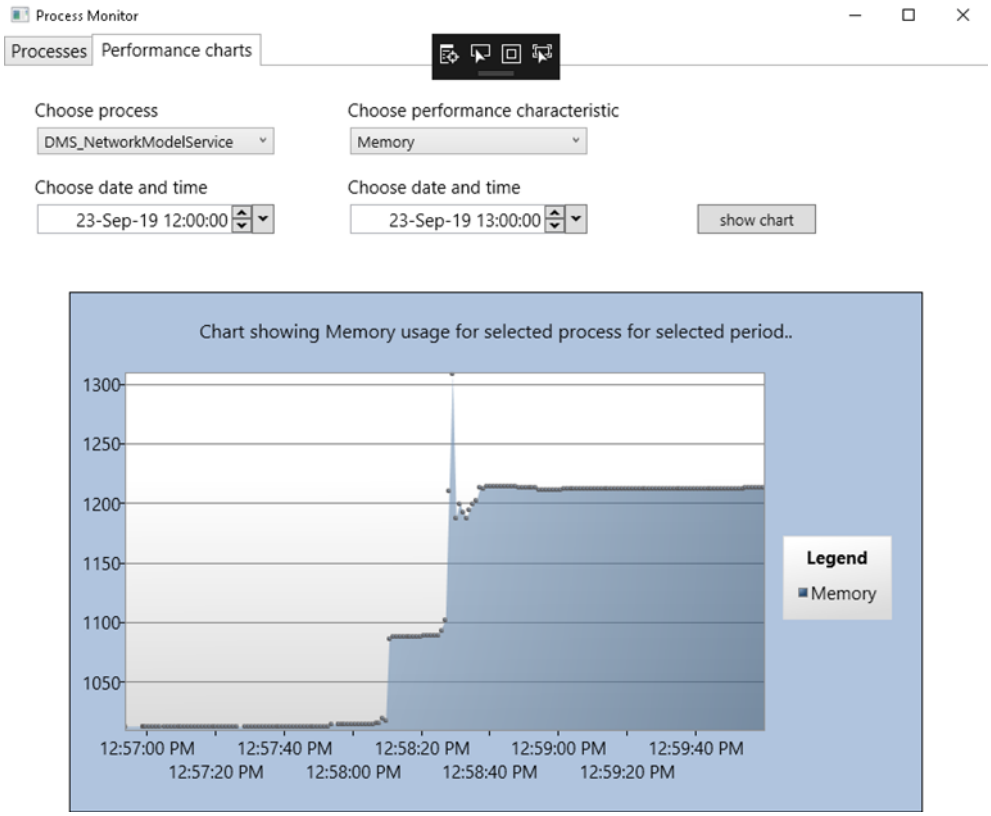
Drugi tab – *Performance charts* korisniku nudi mogućnost kreiranja grafikona za izabrani proces i vremenski period, prikazujući kretanje vrednosti izabrane karakteristike rada (*CPU*, *Memory*, *RAM*) u izabranom vremenskom periodu. Grafički prikaz je često dobar izbor jer korisnici na osnovu njega na intuitivan način mogu doći do zaključaka. Na slici 5 je prikazan tab *Performance charts*. Pri vrhu prozora se nalaze polja putem kojih korisnik vrši izbor parametara za kreiranje grafikona. Polja za izbor vremenskog perioda su posebno detaljna, i korisniku omogućavaju unos tačnog datuma i vremena u formatu HH:mm:ss (sat:minut:sekund format). Nakon klika na dugme *show chart* započinje proces kreiranja grafikona. Kako bi grafikon bio kreiran neophodno je iščitati podatke o radu procesa iz istorijske baze podataka. Jedan primer kreiranja grafikona je dat na slici 5.

2.5 Praćenje izabrane aktivnosti

Softverski alat za nadgledanje rada procesa nudi korisniku posebnu funkcionalnost – povezivanje sa željenim programom i praćenje jedne ili više njegovih aktivnosti. Kako bi se ova funkcionalnost ostvarila neophodno je u manjoj ili većoj meri prilagođenje ovog softverskog alata. Program od interesa, čija se jedna ili više aktivnosti prati, može biti posmatran kao "server" koji izvršava željenu operaciju na zahtev. Aplikacija koja je predmet ovog rada predstavlja "klijenta" koji zahteve inicira, tačnije šalje zahtev "serveru" i pokreće operaciju čije će izvršavanje naknadno pratiti.

Prvi korak u ostvarenju ove funkcionalnosti svakako jeste povezivanje "klijentske" i "serverske" aplikacije, što predstavlja i najveći izazov u njenom ostvarenju. Nakon toga sledi implementacija slanja zahteva za izvršenje određene operacije. Ovaj korak može biti odrađen kao u potpunosti spreman zahtev za slanje, ili da se korisniku ostavi određen nivo slobode da u zahtevu šalje testne podatke koje želi.

Svrha funkcionalnosti praćenja izabrane aktivnosti željenog programa jeste prikupljanje podataka o radu programa u toku izvršenja aktivnosti, kao i kreiranje statistike o njegovom radu u toku izvršenja aktivnosti. Zato je u okviru ovog softverskog rešenja implementirano i kreiranje izveštaja nakon što "serverska" strana završi posao. U izveštaju se nalaze opšti podaci o izvršenoj aktivnosti i grafički prikazi kretanja vrednosti karakteristika rada procesa koji je aktivnost izvršavao. U izveštaj se mogu dodati još neki podaci ukoliko to korisnik želi.



Slika 5. *Primer kreiranja grafikona*

2.6 Primer praćenja izabrane aktivnosti

Kao jedan primer „servera" za potrebe testiranja aplikacije je uzeta aplikacija iz oblasti elektroenergetike, koja služi za prikaz i upravljanje elektroenergetskom mrežom. Operacija koja je bila interesantna za praćenje jeste *Apply Delta*, tačnije operacija koja služi za dodavanje novog modela mreže nad kojom će se raditi.

Komunikacija ostvarena između "klijentske" i "serverske" aplikacije je *WCF* komunikacija. Korisniku je ostavljena sloboda u većoj meri prilikom testiranja ove funkcionalnosti "klijentske" aplikacije time što korisnik bira model/modele mreže koji će biti prosleđeni u sklopu zahteva. Iz tog razloga je bilo neophodno veće prilagođenje aplikacije, kako bi korisniku izbor modela mreže bio dat.

Korisnik bira jedan ili više modela, koji su u formatu CIM/XML fajla. Sledi njihova konverzija u *Delta* format i slanje "serverskoj" aplikaciji. Nakon izvršenja operacije korisniku izlazi *message box* sa osnovnim informacijama o uspešnosti izvršenja operacije i sa mogučnošću kreiranja izveštaja.

Prvi deo izveštaja sadrži osnovne informacije o izvršenju operacije: kada je započela i kada se završila, koliko je trajala, da li je uspešno izvršena, kao i podatke o tome koja *Delta* je primenjena i koliko elemenata je putem nje dodato u mrežu (slika 6).

Date: 10/7/2019

Process name: DMS_NetworkModelService

Operation: ApplyDelta

Delta file name(s): 41.xml,

Total elements inserted: 71

Result type: Succeeded

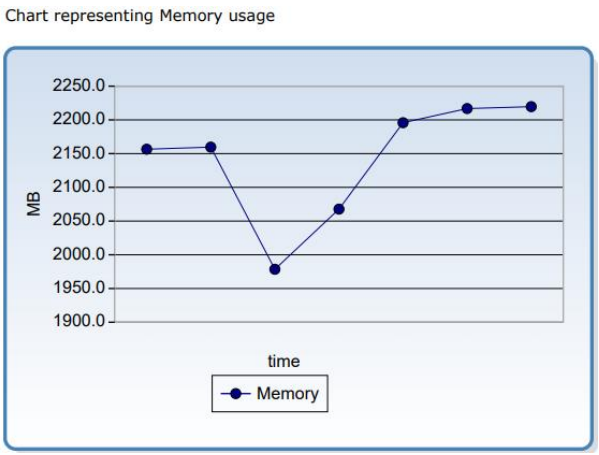
Start time: 10/7/2019 3:53:41 PM

End time: 10/7/2019 3:53:47 PM

Duration: 5.98 seconds

Slika 6. *Statističke informacije iz primera izveštaja*

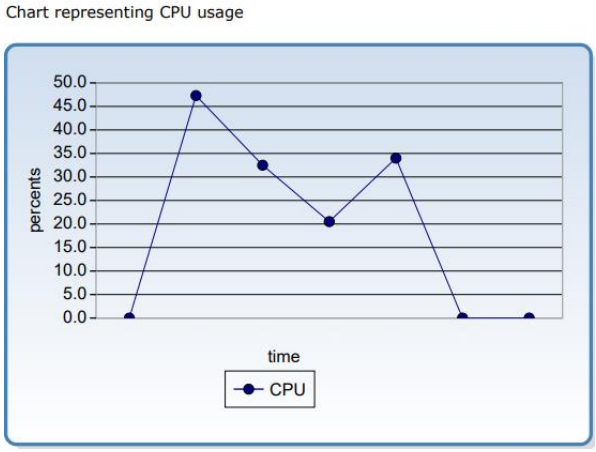
Nakon toga slede grafikoni koji predstavljaju kretanje vrednosti korišćenja *CPU*, *Memory* i *RAM* resursa računara od strane procesa koji je operaciju izvršavao. Izgled grafikona iz izveštaja je dat na slikama 7 i 8.



Slika 7. *Grafikon iz primera izveštaja koji predstavlja iskorišćenje* ***memory*** *resursa računara tokom procesa primene delte*

Na osnovu grafikona prikazanog na slici 7 se vidi da je maksimalna vrednost zauzeća memorije računara od strane procesa: 2220 MB. Početna vrednost zauzeća memorije iznosi: 2150 MB, dok krajnja vrenost iznosi: 2220 MB.

Nakon dodavanja novog modela je zauzeće memorije od strane *Network Model Service* procesa povećano za 70 MB.



Slika 8. *Grafikon iz primera izveštaja koji predstavlja iskorišćenje* ***cpu*** *računara tokom procesa primene delte*

Zaključuje da je maksimalna vrednost zauzeća procesora od strane nadgledanog procesa iznosila 47 %.

3. ZAKLJUČAK

Primene aplikacije za nadgledanje rada procesa su višestruke, počev od čisto informativnog povremenog nadzora, uveravanja u kvalitet proizvoda, preko analize performansi radi izmene arhitekture aplikacije tokom razvoja ili poboljšanja njenog učinka. Mogućnost povezivanja sa drugom izabranom aplikacijom čije operacije mogu biti nadgledane je čini posebnom i dodatno je prilagođava potrebama korisnika. Time dobija širok spektar korisnika. Softverske arhitekte i korisnici koji se bave razvojem svog proizvoda, kao i menadžeri predstavljaju glavnu ciljnu grupu korisnika. Kreiranje izveštaja najviše koristi prilikom dalje analize i izvođenja zaključaka, a izveštaji sačuvani na računaru korisnika predstavljaju jednu vrstu baze podataka.

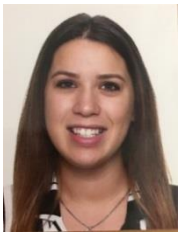
4. LITERATURA

[1] Krishnan M. S., Kriebel C. H., Kekre S., & Mukhopadhyay T. *An empirical analysis of productivity and quality in software products*. Management science, 2000.

[2] Woodside M., Franks G., & Petriu D. C. *The future of software performance engineering*. Future of Software Engineering. IEEE Computer Society, 2007.

[3] Owens M. *The definitive guide to SQLite*. Apress, 2006.

Kratka biografija:



Isidora Ninković rođena je u Novom Sadu 1995. god. Fakultet tehničkih nauka, Departman za Elektrotehniku i računarstvo upisala je 2014. god. Odbrana master rada na Fakultetu tehničkih nauka iz oblasti Elektrotehničkog i računarskog inženjerstva – Razvoj aplikacije za praćenje rada procesa, očekuje se u oktobru 2019. god.

kontakt: isidora_ninkovic@hotmail.com

Зборник радова Факултета техничких наука, Нови Сад

UDK: 004.9
DOI: https://doi.org/10.24867/07BE34Lalic

ЖИВОТНИ ЦИКЛУС ВЕБ АПЛИКАЦИЈА КАО СОФТВЕРСКИХ ПРОИЗВОДА УЗ ОСЛОНАЦ НА АЛАТЕ ЗА CI/CD

LIFE CYCLE OF WEB APPLICATIONS AS SOFTWARE PRODUCTS BASED ON CI/CD TOOLS

Милена Лалић, Бранко Милосављевић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *У раду су анализиране савремене методологије развоја софтвера које укључују и концепте непрекидне интеграције и непрекидне експлоатације. Анализирани су алати који помажу развој веб апликација у складу са овим концептима. У складу са предложеном методологијом је имплементирана веб апликација за резервацију стола у ресторану.*

Кључне речи: *непрекидна интеграција, непрекидна експлоатација*

Abstract – *The paper analyzes the modern methodologies of software development that include concepts of continuous integration and continuous delivery. Tools that improve web application development with these concepts are analyzed. Following the proposed methodology, a web-based restaurant reservation application was implemented.*

Keywords: *continuous integration, continuous delivery*

1. УВОД

Continuous Integration (CI), *Continuous Delivery* (CD) и *Continuous Deployment* (CD) се једним именом називају континуалне праксе које имају за циљ убрзање развоја и испоруке софтверских производа, без смањења квалитета.

Овај рад има за циљ истраживање аутоматизације и усвајање континуалних пракси и алата који омогућавају њену реализацију. Резултати који се добију могу се користити као смернице за повећање свести примене одговарајуће праксе.

2. КОНТИНУАЛНЕ ПРАКСЕ

2.1. Континуална интеграција

2.1.1. „Мрачно“ доба интеграције

Ово поглавље описује мануелни процес и њему својствене недостатке. Описан је пример, који осликава поглед сваког од чланова тима (програмери, тестери и *project manager*).

2.1.2. Шта је континуална интеграција?

Континуална интеграција је пракса која помаже програмерима да испоруче бољи софтвер на поузданији начин. Главна идеја је поделити велике целине, које се развијају, на мање. Битно је да се интеграциони догађају дешавају често. Идеалан случај је да се ово дешава неколико пута на дан, а све то у циљу раног откривања грешака. И последња, али ништа мање важна ставка је процес аутоматског тестирања сваке промене.

2.1.3. Предности коришћења континуалне интеграције

Смањује ризик.

Смањује мануелне процесе који се понављају.

Генерисање софтверског производа који је спреман за испоруку у било ком тренутку.

Већа сигурност у исправно извршавање софтверског производа од стране развојног тима.

2.1.4. Шта спречава тимове да користе CI?

Повећани трошкови одржавања.

Превише промена.

Превише неуспешних build- ова.

Програмери би требали да обављају ове активности.

2.1.5. Праксе и принципи

Континуална пракса може да се уведе у било ком тренутку, међутим она се не односи само на инсталирање алата и њихово коришћење. Заправо, то је потпуно инжењерска пракса која захтева дисциплину.

Управо из тога произилази да је најбоље увести одмах на почетку пројекта и самим тим навикавати чланове тима на нове технологије и развојни процес.

Што чешће интегрисати измене.

Уколико build не прође, потребно је одмах поправити грешку.

Писање тестова.

Сви тестови морају успешно да се изврше.

Локално покренути build.

Не качити код на репозиторијум система за контролу верзија, уколико претходни build није успешно извршен.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био Бранко Милосављевић, ред. проф.

2.2. Континуална пракса

2.2.1. „Мрачно“ доба испоруке

Извођење процеса испоруке је био догађај „великог праска“. Након што се сматрало да је апликација покривена тестовима, неко из тима би добио задатак да запакује апликацију и да је испоручи. Испорука је, такође, веома стресна фаза у циклусу развоја и традиционални приступ је укључивао много мануелних корака. Испорука се дешавала веома ретко, али и дан данас постоје компаније које једном у шест месеци испоруче нову верзију.

2.2.2. Шта је континуална испорука?

Континуална испорука је пракса паковања и припреме апликације за продукцију, што је чешће могуће. Континуална испорука води континуалну интеграцију корак даље. Свака нова функционалност је потенцијални кандидат за пуштање у продукцију. Опет зависно од организације, али у суштини да ли ће одређена функционалност ићи у продукцију или неће, захтева људску интервенцију. Једном када је CD постављен, процес испоруке постаје тривијалан, јер се све може обавити притиском једног дугмета.

2.3. *Continuous deployment*

Овај развојни приступ надограђује континуалну испоруку и у потпуности уклања људску интервенцију. Сваки кандидат за *release* за који се сматра да је спреман (тј. сви тестови пролазе) се аутоматски пушта у продукцију.

2.3.1. *Continuous delivery* и *continuous deployment*

Continuous delivery обухвата низ пракси које обезбеђују да се имплементација софтверског производа може пустити у продукцију на брз и сигуран начин тако што ће се вршити испорука сваке измене у окружење које је слично продукцијском. Пуштање измена у продукцију се врши притиском на једно дугме.

Continuous deployment је следећи корак након *continuous delivery* фазе. Свака измена која прође аутоматизоване тестове се аутоматски пушта у продукцију.

Поента је, на основу пословних потреба, одлучити да ли је *continuous deployment* управо оно што је компанији неопходно. Насупрот томе, *continuous delivery* је апсолутно неопходна, јер све измене могу да се испоруче клијенту кликом на једно дугме.

3. АЛАТИ ИМПЛЕМЕНТИРАНОГ РЕШЕЊА

3.1. Систем за контролу верзија

Примена континуалне интеграције је незамислива без система за контролу верзија. Заправо, тешко је замислити процес израде софтверског производа без њега.

Систем за контролу верзија управља променама над дигиталним артифактима (најчешће фајловима и фолдерима). Постоје две архитектуре оваквих система. Код централизованих система сва историја се налази у репозиторијуму централног сервера, а клијенту имају само текућу верзију.

Представници су *SVN, Perforce, Subversion.* Код дистрибуиране архитектуре, клијент има увид и пуну историју измена. Представници су *Git, Mercurial* и *Bazaar.*

3.1.1. *Git*

Git је дистрибуирани систем за контролу верзија. *Git* о подацима размишља као о скупу снимака (*snapshot*- а) минијатурног фајл система.

Репозиторијум је скуп метаподатака.

Радни директоријум је локални фолдер у коме се мењају и креирају фајлови који су обухваћени верзионирањем.

Простор припреме фајл који чува информације о томе шта ће бити део следећег трајног бележења.

Удаљени репозиторијум је место где се чувају фајлови пројеката на неком серверу.

HEAD је замишљени показивач.

Команда add додаје фајл у локални репозиторијум и припрема га за *commit.*

Команда commit прави снимак промена стања фајлова у односу на претходно.

Командом push се шаљу измене из локалног репозиторијума на удаљени репозиторијум.

Командом fetch се повлачи последње стање са удаљеног репозиторијума.

Грана је алтернативни ток развоја.

3.1.2. *GitLab*

GitLab је систем за управљање *git* репозиторијумима. Софтвер је креирао *Dmitriy Zaporozhets* 2011. године. Првобитно је написан у програмском језику *Ruby*, а касније су неки делови пребачени у програмски језик *Go.*

3.2. *Jenkins*

Jenkins је сервер отвореног кода континуалне интеграције који оркестрира низом акција на аутоматизован начин. Предност у односу на друге алате је та што нуди велики број додатака.

3.2.1. Основни појмови *Jenkins* сервера

Jenkins Pipeline је група задатака који су међусобно повезани. Задаци се извршавају један за другим, односно након успешно извршеног једног задатка прелази се на други.

Корак (step) је основни део *pipeline*- а и говори *Jenkins*- у шта да уради.

Jenkinsfile је текстуални фајл у ком се врши дефинисање *pipeline*- а.

Чвор (node) је основни концепт *scripted pipeline*- а.

3.3. *Sonatype Nexus*

Nexus је менаџер складишта. Омогућава посредан приступ централном репозиторијуму, прикупљање и управљање зависностима.

Централни репозиторијум је скраћено од *Central Maven Repository (CMR)* и може се посматрати као глобални менаџер репозиторијума који чува све компоненте отвореног кода.

3.4. *Docker*

Docker је алат отвореног кода који је дизајниран да олакша креирање, *deploy* и покретање апликације. Заснован је на идеји паковања кода апликације са

свим зависностима у преносиву јединицу која се зове контејнер. Захваљујући томе, програмери могу бити сигурни да ће апликација радити на било којој машини.

3.4.1. *Docker* слика

Docker слика је фајл, који се састоји од више слојева. Контејнер се стартује на основу слике. Слике се чувају у *Docker* регистру, као што је *DockerHub* и могу се преузети *docker pull* командом.

3.4.2. *Dockerfile*

Dockerfile је текстуални документ који садржи све наредбе које би се обично ручно извршавале да се изградила *Docker* слика. Свака команда ствара један свој слике.

FROM. Прва команда. Дефинише базну слику.

EXPOSE. Отвара задати порт.

RUN. Омогућава инсталацију апликација и пакета који су неопходни.

ENTRYPOINT. Омогућава конфигурацију контејнера.

4. ИМПЛЕМЕНТИРАНО РЕШЕЊЕ

4.1. *NoWait* апликација

Пројектни задатак представља апликација која омогућава корисницима да резервишу сто у жељеном ресторану у одређеном термину. Резервација је омогућена у различитим ресторанима који су регистровани у оквиру система.

4.2. Процес рада имплементираног решења

Програмери, локално на својим машинама пишу код за одређену функционалност. Оног тренутка када програмери покрену команду *push* система за контролу верзија, сав код се чува на репозиторијуму. На *GitLab*- у је подешен *webhook* који ће обавестити *Jenkins* да се десила измена. *Jenkins* на основу *Jenkinsfile*- а извршава задатке.

5. ЗАКЉУЧАК

Овај рад представља увод у нове термине и праксе које пре или касније чекају развој софтверских производа. Постављена архитектура обезбеђује израду континуалних пракси, које својом толеранцијом на промене, усмеравају и контролишу развојни процес. Дато решење обезбеђује рано откривање грешака, њихово брзо уклањање и побољшање квалитета.

Недостаци који захтевају будућа истраживања:

Vagrant – алат за брзо и једноставно подешавање комплетног *development* окружења виртуелне машине. Једноставним начином за коришћење и фокусом на аутоматизацију, смањује време за подешавање радног окружења.

Kubernetes - систем за покретање и координацију контејнерских апликација. Платформа је дизајнирана за потпуно управљање животним циклусом контејнерских апликација и услуга користећи методе које пружају предвидљивост, скалабилност и велику доступност.

6. ЛИТЕРАТУРА

[1] Before CI/CD, https://thenewstack.io/understanding-the-difference-between-ci-and-cd/

[2] Paul Duvall, Steve Matyas, and Andrew Glover, Continuous Integration Improving Software Quality and Reducing Risk,

[3] Lauri Hukkanen, Adopting Continuous Integration – A Case Study, Aalto University, 2015.

[4] Continuous integration vs delivery vs deployment, https://www.atlassian.com/continuous-delivery/principles/continuous-integration-vs-delivery-vs-deployment

[5] Manoj Mahalingam S, Learning Continuous Integration with TeamCity, Packt Publishing, 2014.

[6] Игор Дејановић, Системи за контролу верзија - увод, слајдови са предавања, Универзитет у Новом Саду, Нови Сад, 2016., http://www.igordejanovic.net/courses/tech/sistemi-za-kontrolu-verzija.html#1

[7] Scott Chacon, Ben Straub, Pro Git, apress, 2014.

[8] Игор Дејановић, *Git DVCS*, слајдови са предавања, Универзитет у Новом Саду, Нови Сад, 2018., http://www.igordejanovic.net/courses/tech/git.html#1

[9] Git pull, https://www.atlassian.com/git/tutorials/syncing/git-pull

[10] About Jenkins, https://www.guru99.com/jenkin-continuous-integration.html

[11] Jenkins Pipeline, https://jenkins.io/doc/book/pipeline/

[12] Syntax Comparison, https://jenkins.io/doc/book/pipeline/syntax/#compare

[13] Sonatype Nexus, https://blog.sonatype.com/2010/04/why-nexus-for-the-non-programmer/

[14] Jeff Nickoloff, Docker in Action, Manning Publications Co., 2016

Биографија

Милена Лалић је рођена 19.6.1994. године у Книну, Хрватска. Основну школу „Михајло Пупин“ је завршила 2009. године у Ветернику. Гимназију „Исидора Секулић“ завршила је у Новом Саду, 2013. године. Исте године је уписала Факултет техничких наука, смер Рачунарство и аутоматика. Основне академске студије завршава 2017. године и исте године уписује мастер академске студије из области Рачунарство и аутоматика, Примењене рачунарске науке и информатика. У септембру 2018. године је положила последњи испит и тиме стекла услов за одбрану мастер рада. Тренутно ради као *software developer* у фирми *Pannonia-Expertise.*

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 007:005
DOI: https://doi.org/10.24867/07BE35Zekovic

IMPLEMENTACIJA REACT-TABLEQL BIBLIOTEKE OTVORENOG KODA ZA TABELARNI PRIKAZ PODATAKA

IMPLEMENTATION OF REACT-TABLEQL OPEN SOURCE LIBRARY FOR TABULAR VIEW OF DATA

Danilo Zeković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu predstavljena je biblioteka otovrenog koda React-TableQL, koja služi da se podaci pribave i prikažu u formi tabele. Biblioteka sadrži dve komponente koje uz određenu konfiguraciju dodaju ili modifikuju različite funkcionalnosti tabele. Podaci se nabavljaju pomoću GraphQL upita koji se prosledi Apollo Client-u. Za prikaz tabele i obradu podataka korišten je programski jezik JavaScript i biblioteka React.*

Ključne reči: *Biblioteka, React, tabela, GraphQL*

Abstract – *This paper presents the open source library React-TableQL, which is used for fetching and displaying data in a form of a table. Library contains two components, which with certain configuration add or modify different functionalities of the table. Data is fetched using GraphQL query that is passed to Apollo Client. JavaScript and react are used for display and parsing of data.*

Keyords: *Library, React, table, GraphQL*

1. UVOD

U dvadeset i prvom veku, usled intenzivnih društvenih promena, ubrzanog života i transformacije radnih uslova, izuzetno bitan aspekt poslovanja predstavlja brzina i efikasnost u radu zaposlenih, i to ne samo u proizvodnji potrošne robe ili prehrambenih proizvoda, već i u razvoju softvera. Frederick P. Brooks piše u svojoj knjizi "The Mythical Man Month" kako predviđanje trajanja razvoja jednog softverskog sistema predstavlja jedan od težih i često pogrešno izvedenih koraka u samom razvoju. On ističe da su programeri često previše optimistični i da u tom optimizmu prave greške u predviđanjima. Takođe, naglašava da povećanje broja ljudi koji su angažovani na istom projektu najčešće ima negativan efekat na krajnji rezultat i trajanje razvoja samog softvera. Prema Brooksovim tvrdnjama, kompleksnost nekog softvera programeru deluje mnogo jednostvanija nego što ona u stvarnosti zaista jeste, i u tom neslaganju dolazi do velikog gubitka u vremenu i novcu [1].

Jedan od najčešćih načina prikazivanja informacija je u vidu tabele. Kao takav postoje mnoge varijacije i funkcionalnosti koje bivaju dodate samoj komponenti proširujući njenu kompleksnost.

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

Sa porastom kompleksnosti raste i vreme utrošeno na implementaciju, a potom i na ažuriranje, odnosno održavanje same komponente. Da bi se neki podaci prikazali u tabeli potrebno je prvo doći do njih i to se na vebu najčešće vrši putem HTTP/HTTPS poziva. Zahvaljujući inžinjerima iz Facebook-a[1] danas, pored REST[2] poziva, imamo mogućnost da koristimo GraphQL za dobavljanje i manipulaciju podacima. GraphQL[3] sve više raste u popularnosti i sama njegova primena je u usponu.

2. TEHNOLOGIJE

TableQL je konstruisan od dve ključne tehnologije i dve biblioteke koje su izvedene iz njih. Dve ključne tehnologije su JavaScript i GraphQL. Prva biblioteka koja je korištena, a napisana je JavaScript programskim jezikom, je React. Druga je Apollo[4] biblioteka koja je napisana JavaScript jezikom da bi se GraphQL lakše uvezao i koristio u veb aplikacijama.

2.1 JavaScript

JavaScript je programski jezik kreiran od strane Brendana Eich-a [2]. Prvi put je predstavljena 1995. godine, kao programski jezik za pisanje programa za Netscape Navigator pretraživač (browser). Međutim, posle sporijih početaka, svi vodeći pretraživači su usvojili JavaScript kao programski jezik koji se koristi da veb stranicama i aplikacijama omogući interaktivnost i funkcionalnost bez potrebe da se stranica ažurira posle svake izvršene akcije korisnika [3].

Posle usvajanja JavaScript kao programskog jezika od strane pretraživača, kreiran je standard poznat kao ECMAScript[5] standard. Organizacija pod nazivom Ecma International je sastavila standard i do današnjeg dana objavljeno je više verzija pod njihovim imenom [3]. JavaScript je postao popularniji u periodu od 2000. do 2010. godine sa verzijom ECMAScript 3, poznatiji pod skraćenim imenom ES3. 2008. godine planirane su radikalne izmene za verziju ES4, međutim verzija 4 nikada nije bila puštena zbog svojih drastičnih izmena u samom jeziku. Umesto verzije ES4, iste godine puštena je verzija ES5 sa manjim izmenama u odnosu na ES3, i na taj način je još više podigla popularnost JavaScript-a. Posle verzije ES5 usledila je verzija ES6 u 2015. godini, i

[1] https://www.facebook.com
[2] https://restfulapi.net/
[3] https://graphql.org/
[4] https://www.apollographql.com/
[5] https://www.ecma-international.org/

od tad je svake godine izlazila nova verzija sa manjim ili većim izmenama [2].

2.2 React

React je JavaScript biblioteka otvorenog koda za kreiranje korisničkog interfejsa. Inicijalno je pušten u javnost za upotrebu 2013. godine od strane tehnološkog giganta Facebook-a i do danas je održavan i razvijan od strane Facbook-a i open source zajednice. Od tada React tim je radio i stvorio mogućnosti da se React koristi i na mobilnim uređajima pomoću React Nativ-a[6], i da može pomoću Node.js-a[7] da se vrši generisanje na serverskoj strani. Pored raznovrsnosti primene, najveća prednost React-a je da se brzo i lako uči. React se zasniva na komponentama koje su osnovni deo React aplikacije [4].

2.3 GraphQL

GraphQL je jezik za upite (query language) za API. Drugim rečima, GraphQL je set pravila za opis podataka koje prednja strana potražuje od zadnje strane, servera. GraphQL omogućava programerima da opišu podatke koji su im potrebni, i da ne brinu o tome kako će upit biti poslat i podaci da se dobave. Uglavnom se koristi preko HTTP poziva, iako nije ograničen na HTTP [5].

Prednosti GraphQL-a u odnosu na REST API je da sprečava dovlačenje više ili manje podataka u zavisnosti od situacije. Ponekad kad pošaljemo zahtev za korisnika, sve što nam je potrebno su ime i datum rođenja. REST će vratiti celog korisnika, ne samo ime i datum rođenja, već i boju očiju, mesto rođenja, ime majke i tako dalje. Sa druge strane, GraphQL nam omogućava da tražimo tačno šta nam treba vratiće nam samo to što je traženo. U drugom slučaju, imamo situaciju gde nije dovoljno podataka dobijeno prilikom prvog zahteva. Pretpostavimo da je korisnik sportista i da je igrao u više timova. Potreba aplikacije je da se ispišu nazivi svih timova za koje je igrao. Timovi za koje je igrao su vezani za korisnika preko njihovih indentifikacionih brojeva, i da bi došli do njihovih imena morali bismo da pozovemo API onoliko puta koliko ih ima, što stvara nepotreban broj poziva u slučaju da se koristi REST. GraphQL taj problem rešava tako što dozvoljava da se podaci opišu ugneždenim upitima.

Još neke prednosti GraphQL-a su da GraphQL ima samo jednu putanju koju gađa za dobavljanje podataka. Ta putanja komunicira preko HTTP POST metode, kojoj prosleđuje objekat sa svim potrebnim opisima podataka koji su klijentu potrebni. Na taj način rešen je problem nekontrolisanog broja putanja.

2.4 Apollo Client

Apollo Client je JavaScript biblioteka otvorenog koda za upravljanje stanjem. Omogućava da programer napiše GraphQL zahteve, a Apollo Client se pobrine za pravljenje upita i keširanje podataka, kao i za ažuriranje korisničkog interfejsa. Apollo Client ima odvojene biblioteke za različite JavaScript radne okvire, među kojima je React biblioteka najpopularnija. Postoje tri osnovne komponente i funkcionalnosti Apollo Clienta: *query*, *mutation*, i upravljanje lokalnim stanjem [6].

[6] https://facebook.github.io/react-native/

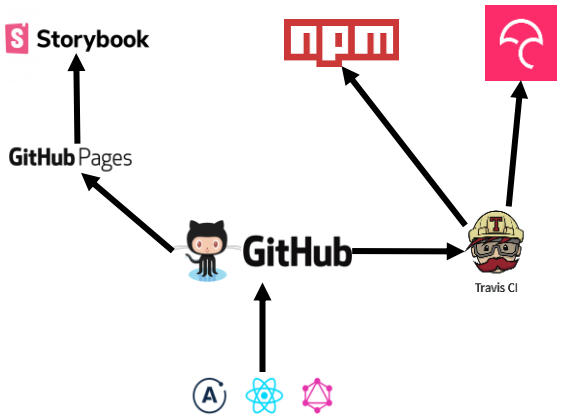
[7] https://nodejs.org/en/

3. REACT-TABLEQL

React-tableql koja je predmet ovog master rada je JavaScript React biblioteka. Napisana je i objavljena kao open source (otvoren kod) projekat sa ciljem da dosegne što šire krugove među React programerima. Svrha biblioteke je da omogući brži i efikasniji razvoj React aplikacija omogućavajući da se sa nekoliko redova koda doda tabela sa svim kompleksnim funkcionalnostima koje može da poseduje. Takođe, za cilj ima i da se omogući da se što manje rizika i grešaka pravi tokom razvoja iste.
TableQL je komponenta koja predstavlja samu srž react-tableql biblioteke. Ona omogućava tabelarni prikaz podataka, u skladu sa definisanom konfiguracijom. Međutim, to ne čini ovu biblioteku drugačijom od drugih već postojećih generatora tabelarnog prikaza podataka. Ono što izdvaja react-tableql od drugih sličnih biblioteka i komponenti je da sadrži omotač oko same komponente za prikaz podataka koji koristi GraphQL da napravi zahtev za podacima serveru koji ima definisan GraphQL API, sa dodatinim slojem, koji pruža dodatni nivo apstrakcije i samim tim smanjuje prostor za grešku prilikom razvoja. Komponenta koja sadrži spoljašnji omotač koji je zadužen za GraphQL API komunikaciju je ApolloTableQL.

3.1 Organizacija i struktura

React-tableql kao open source projekat ima svoj *source code* javno dostupan. Kod se čuva i može se videti kao i doprineti njegovom razvoju na GitHub-u. GitHub, kao centralna komponenta u arhitekturi projekta, igra vezivnu ulogu između napisanog koda i njene isporuke široj javnosti, kao i pristupačnost same dokumentacije.



Slika 1. *Struktura projekta*

Dokumentacija je veoma bitan sastavni deo svakog softvera. Kako bi programeri znali da koriste react-tableql biblioteku, velika pažnja je posvećena pisanju dokumentacije i njenoj dostupnosti. Pored dokumentacije koja je napisana u sklopu samog projekta u radnom dokumentu, kreirana je Storybook[8] prezentacija funkcionalnosti biblioteke. Sadržaj readme fajla može se pročitati na GitHub-u kao i na sajtu www.npmjs.com/package/react-tableql. Sa druge strane imamo GitHub Pages koje nam omogućavaju da direktno iz GitHub repozitorijuma imamo sajt za projekat. GitHub Pages su integrisane sa GitHub-om, i u konfiguraciji projekata postoji mogućnost da se servira veb stranica koja služi kao veb prezentacija projekta. URL stranice je automatski sastavljen od naziva projekta i korisničkog imena vlasnika projekta, što može

[8] https://storybook.js.org/

biti osoba ili organizacija. U slučaju react-tableql biblioteke URL glasi https://danilo-zekovic.github.io/react-tableql.
Prilikom svakog doprinosa biblioteci korišćenjem git-a i komitovanja koda na GitHub-u okida se kontinuirana integracija. Za kontinuiranu integraciju korišćen je Travis CI[9]. Travis CI se lako integriše sa GitHub-om i automatski se pokreće preko GitHub-a. Prilikom svakog pokretanja validiraju se različite stvari koje je moguće definisati kroz konfiguracioni dokument. Taj dokument je deo projekta i Travis CI ga automatski prepoznaje jer konvencija nalaže da se zove .travis.yml [7].
U sklopu react-tableql biblioteke proverava se ispravnost sledećih uslova:

1. da li svi testovi prolaze bez javljanja grešaka,
2. kolika je pokrivenost koda testovima,
3. da li je napisani kod pravilno formatiran,
4. da li se biblioteka i Storybook stranica grade (build) bez grešaka

Isporuka se vrši na npm. Npm je paket menadžer za JavaScript. Svaki paket ima jedinstveno ime preko kojeg je indentifikovan. ReactTableQL biblioteka takođe ima svoje ime u npm registru, react-tableql. Jednostavno se može dodati u projekat korišćenjem komande *npm install react-tableql* u konzoli. Međutim, da bi mogla da se instalira poslednja stabilna verzija biblioteke prvo mora da se sadržaj paketa doda u npm registar. Proces dodavanja u registar je automatizovan korišćenjem Travis-a. Nakon sto Travis završi svoje provere i izgradnju modula oni se šalju na npm uz pomoć API ključa koji identifikuje datu pošiljku kao validnu i kod postaje dostupan preko npm konzole. Slanje koda u registar se ne dešava svaki put kad je kod komitovan na GitHub i kad prođe provere na Travis-u. To se dešava samo kad se tag verzije promeni. Tag verzije se čuva u *package.json* dokumentu koji je izuzetno bitan za svaki projekat koji koristi npm. U njemu se čuvaju svi detalji vezani za projekat i pakete koji se koriste u njemu. Na primer, u react-tableql biblioteci package.json dokument sadrži podatke o verziji biblioteke, podatke o autoru, nazive i verzije paketa koji su potrebni za funkcionisanje biblioteke, kao i komande za pokretanje projekta.

3.2 Funkcionalnosti i primena

React-tableql biblioteka sadži dve komponente, TableQL i ApolloTableQL. Obe komponente imaju svoje parametre koji omogućavaju da se tabela konfiguriše na različite načine. Većina parametara su indentični za obe komponente, međutim, ima ih nekoliko po kojima se razlikuju.

Raznovrsnost parametara ima za cilj da omogući veću prilagodljivost i fleksibilnost korisnikovim potrebama. Komponente imaju svoju osnovnu konfiguraciju koja zadovoljava potrebe jedne aplikacije, ali kako su aplikacije različite, tako su i potrebe za različitim komponentama potrebne. Iz tog razloga težilo se ka omogućavanju korisniku biblioteke da promeni i podesi svaki aspekat komponenti prema svojim potrebama.

Još jedan aspekt koji je bio motivacija za razvoj biblioteke je da se kod brže i efikasnije razvija. To je ostvareno tako što je kompleksnost funkcionalnosti sakrivena. Programer piše manje koda a dobija kompleksne rezultate. Na slici 2 moguće je videti minimalno potreban broj linija koda da bi se dobila tabela pomoću ApolloGraphQL komponente:

```
import React from 'react'
import { ApolloTableQL } from 'react-tableql'

const ExampleComponent = () => {
  const GET_ALL_FILMS = `
    {
      allFilms {
        films {
          title
          episodeID
        }
      }
    }
  `
  return <ApolloTableQL query={GET_ALL_FILMS} />
}

export default ExampleComponent

```

Slika 2 Primer minimalnog koda za generisanje tabele

Pre nego što se tabela konfiguriše, potrebno je dodati react-tableql u projekat. Dodavanje biblioteke je jednostavno i može se ostvariti jednom komandom u *root* folderu projekta, *npm install --save react-tableql*. Da bi se ta komanda pokrenula, neophodno je prethodno instalirati npm u konzoli. Navedena komanda sadrži dodatnu oznaku *--save*, koji dodaje react-tableql u *packege.json* projekat u kojem je pokrenuta. Komanda će iz npm registra svući poslednju stabilnu verziju paketa i dodati je projektu. U tom trenutku su komponente spremne da se učitaju u kod i koriste. Neke od najbitnijih parametara konfiguracije su *query*, *columns* i *pagination*.

Query je jedini obavezan parametar ApolloTableQL komponente. Tip vrednosti *query*-ja je string, koji predstavlja GraphQL upit. Ovaj parametar je ključan jer od njega zavisi koji će sepodaci dovući sa servera. Nakon što se string koji predstavlja upit prosledi ApolloTableQL komponenti, ona prosledi tu vrednost funkciji *gql* koja se učita iz biblioteke graphql-tag. Graphql-tag biblioteka omogućava da se string raščlani i pretvori u validan GraphQL upit [8]. Nakon što se formira ispravan format upita, on se prosleđuje Apollo Client komponenti *Query*, koja pored parametra koji sadrži upit prima i mnoge druge o kojima će biti više reči kasnije u radu. Nakon što je prosleđen ovaj parametar, vrši se upit ka GraphQL API-ju, a u isto vreme se varijabla *loading*, što na engleskom znači učitavanje, prosleđuje TableQL komponenti da se naznači da se podaci čekaju na učitavanje. API vraća podatke u obliku niza koji se zatim prosleđuju TableQL komponenti. U trenutku kad se prime podaci i prosledi *loading* parametar, menja se vrednost u *false* i naznaka učitavanja prestaje. Podaci se obrađuju i bivaju prikazani u tabeli.

Columns predstavlja parametar koji omogućava detaljnije izmene i prilagođavanje komponente specifičnim zahtevima. *Columns* nije obavezan parametar. *Columns* može biti niz stringova i objkata, mogući parametri objekta su *id*, *label*, *component*, *customolumn*, *headerStyle*, *nodeStyle* i *sort*. Od svih parametara jedino je *id* obavezan kako bi bilo moguće indentifikovati i povezati kolonu sa podacima koji su vraćeni sa servera.

React-tableql ima opciju da se uključi paginacija, kao i da se konfiguriše prema potrebama koje aplikacija iziskuje. Opcija da se prosledi jedinstvena paginacija još uvek ne postoji, iako je u planu da u nekim budućim verzijama

[9] https://travis-ci.org/

bude dodata. *Pagination* je opciono svojstvo, koje prima dva tipa vrednosti *boolean* i objekat sa svojstvima *pageLimit*, *pageNeighbors*, *currentPage*, *onPageChange* i *style*.

4. POSTOJEĆA REŠENJA

Tabela je jedan od najpopularnijih vidova prikaza podataka, i kao takav ima i mnoga rešenja za generisanje i prikaz za različite platforme. React je veoma rasprostranjen među veb aplikacijama i samim tim ima široku ponudu komponenti koje omogućavaju prikaz kompleksnih tabela na jednostavan način. Mnoga rešenja su deo biblioteka koje pružaju i druge komponente za izgradnju korisničkog interfejsa, kao na primer, Material Design [10] ili Semantic UI [11]. Samim tim što sadrže mnogo više od jedne komponente, kompleksnost te komponente je svedena na minimum i funkcionalnosti koje pruža su ograničene. Sa druge strane postoji biblioteka react-table [12], čiji fokus je isključivo na tabelarnom prikazu podataka.

React-table je React komponenta za prikaz kompleksnih tabela. React-table je brz i prilagodljiv sa mnogo testiranih funkcionalnosti. Neke od funkcionalnosti koje poseduje su: sortiranje, filtriranje, paginacija i još mnoge druge. Orginalno je kreirana biblioteka od strane Tanner Linsley-a.

5. ZAKLJUČAK

React-tableql je JavaScript biblioteka koja pruža korisniku mogućnost da na lak i jednostavan način prikaže podatke u tabeli. Ovaj projekat napravljen je kako bi se našao način da se ubrza razvoj aplikacija, a u isto vreme smanji obim koda koji treba da se napiše. U radu je opisano rešenje koje ujedinjuje više različitih tehnologija otvorenog koda u jednu jedinstvenu biblioteku. Biblioteka pruža dve React komponente koje uz minimalno podešavanja omogućavaju da se napravi upit za podatke korišćenjem GraphQL-a, a potom i prikažu u stilizovanom i unapred podešenim funkcionalnostima tabele.

TableQL i ApolloTableQL su za sada jedine komponente koje čine react-tableql. One pružaju raznovrsne funkcionalnosti poput paginacije, sortiranja i automatizovanog parsiranja podataka bez prethodnog podešavanja od strane korisnika. Osnovna prednost korišćenja komponenti iz biblioteke opisane u ovom radu jeste da one omogućavaju bezbedniji i brži razvoj aplikacija. To je postignuto kroz apstrakciju funkcionalnosti koje jedna tabela može da ima. I pored visokog nivoa apstrakcije, ostavljen je prostor za prilagođavanje svakog aspekta komponenti.

7. LITERATURA

[1] Brooks, Frederick. *The Mythical Man-Month*. Addison-Wesley, 1995

[2] Frost, Aaron. *JS.next: a Manager's Guide*. O'Reilly Media, 2015.

[3] Haverbeke, Marijn. *Eloquent Javascript: a Modern Introduction to Programming*. No Starch Press, 2018.

[4] "React – A JavaScript Library for Building User Interfaces." – *A JavaScript Library for Building User Interfaces*, https://reactjs.org/.

[5] Porcello, Eve, i Alex Banks. *Learning GraphQL: Declarative Data Fetching for Modern Web Apps*. O'Reilly Media, 2018.

[6] "Apollo Docs." *Apollo*, https://www.apollographql.com/docs/react/.

[7] "Travis CI - Test and Deploy Your Code with Confidence." *Travis CI* , https://travis-ci.org/.

[8] "Graphql-Tag." *Npm*, https://www.npmjs.com/package/graphql-tag.

Kratka biografija:

Danilo Zeković rođen je 11.05.1993 u Novom Sadu. Osnovnu školu "Petefi Šandor" u Novom Sadu završava 2008. godine kao dobitnik Vukove diplome. Iste godine upisuje gimnaziju "Svetozar Marković" u Novom Sadu, opšti smer. Četvrtu godinu srednje škole upisuje u Munster High School u Sjedinjenim Američkim Državama u mestu Munster, Indijana. 2012. završava srednju školu uz počasti za odličan prosek. Potom upisuje Saint Joseph College u Rensselaer, Indijana. Diplomira u maju 2016. godine na smeru Computer Science. 2017. godine upisuje master studije na smeru Softversko inžinjerstvo i informacione tehnologije. Poslednji ispit položio januara 2019. godine. Prosek položenih ispita na master studijama je 9.83.

[10] https://material-ui.com/
[11] https://react.semantic-ui.com/
[12] https://www.npmjs.com/package/react-table

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE36Zivlak

PROGRAMSKO REŠENJE ZA RAZVOJ MEĐUSCENA JUNITI POGONA

SOFTWARE SOLUTION FOR UNITY INTERSCENE DEVELOPMENT

Stevan Zivlak, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – *Cilj ovog rada jeste pronalazak rešenja problema razvoja međuscena korišćenjem Juniti pogona. U radu se koriste C# skripte da bi se aktivirale međuscene u ključnim trenucima, kao i za sinhronizaciju trenutaka izvršavanja događaja unutar njih. Skripte se oslanjaju na interne Juniti mehanizme za detekciju kolizija. Ova solucija pruža velik stepen kontrole, ali postoje bolja rešenja kada je u pitanju brzina razvoja.*

Ključne reči: *Međuscene, razvoj video igara, Juniti pogon*

Abstract – *This paper aims to present a solution for the problem of cutscene development using Unity engine. C# scripts are used to listen for key game events, trigger cutscenes when such events occur and synchronize the order and timing of events within them. The scripts use internal Unity mechanisms for collision detection. This solution gives unprecedented control, but better solutions exist in terms of development speed.*

Keywords: *Cutscenes, game development, Unity engine*

1. UVOD

Većina video igara današnjice ima u sebi međuscene – delove igre sa smanjenim nivoom interakcije koji blokiraju gejmplej. One se obično koriste u svrhu proširivanja narativa igre, ili kao oruđe za izazivanje određenih emocija kod igrača kao što je osećaj gubitka kontrole i straha.

Razvoj međuscena je po svojoj prirodi problem automatizacije dešavanja događaja. Uključuje kontrolu kretanja modela u igri i kontrolu animacija. Dodatno, potrebno je kontrolisati i zvučne i specijalne akcione efekte. Na kraju, svi pomenuti elementi moraju se vremenski sinhronizovati među sobom.

Pronalazak adekvatnog programskog rešenja za razvoj međuscena veoma je bitno za postizanje balansa između brzine razvoja i mogućnosti kustomizacije koja je potrebna. Izbor Juniti pogona kao razvojnog okruženja motivisano je delimično tehničkim mogućnostima koje nudi, a delimično ogromnom rasprostranjenošću istog u nezavisnom razvoju video igara [1].

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

2. GRADIVNI ELEMENTI MEĐUSCENA

Međuscene sadrže vizuelne i zvukovne gradivne elemente koje se prikazuju na ekranu. Od ključnog je značaja da su ovi elementi konzistentnog stila da bi komunikacija sa igračom bila efektna.

2.1. Vizuelni gradivni elementi

U prošlosti je 2D animacija u video igrama pravljena tako što bi se ručno crtao svaki frejm. Problem kod ovog pristupa je što je potrebno nacrtati ogroman broj slika za svaku animaciju (igre tipično imaju od 30 do 60 frejmova u sekundi), a treba i uložiti napor da se obezbedi da je prelaz između svaka dva frejma konzistentan. S druge strane, ovaj pristup omogućava da se art tim više posveti svakom frejmu, pogotovo senčenju, što daje finalnom proizvodu iluziju dubine koju imaju 3D modeli.

Slika 1. *Vektorski crtež*

Moderna kompjuterska animacija se oslanja na vektore i interpolaciju da bi smanjila količinu posla koji treba da se obavi. Na slici 1 (levo) se može videti izgled jednog neobojenog vektorskog crteža u programu za kreiranje vektorske grafike Adobe Illustrator CC. On je skup tačaka i linija koje ih povezuju.

Ono što je posebno kod ovakvih crteža jestee što je svaki deo lika moguće selektovati i nezavisno pomerati ili menjati. U sredini slike se može videti originalan izgled istog vektora u obojenoj varijanti, a desno je primer izmene gde je leva noga rotirana, brkovi su obrisani, a svetlo na kacigi je promenilo boju.

Na svakom delu vektorskog crteža moguće je izvršiti rotaciju, translaciju ili promenu veličine bez ikakvog dodatnog crtanja. Na mestima na neobojenom crtežu gde su tačke moguće ih je povući i izmeniti oblik okolnih linija, a moguće je i dodati nove tačke na mestu na kome ih nema. Takođe još jedna pogodnost vektora jeste što, iako su neki delovi crteža iznad nekih drugih (na primer brkovi prekrivaju deo lica), ni jedan deo se ne briše crtanjem preko njega, već se samo prekriva.

Slika 2. *Samo pet ključnih frejmova za animaciju*

Na slici 2 prikazana je animacija umiranja karaktera koja je napravljena korišćenjem originalnog vektora. Pet prikazanih slika nisu klasični frejmovi već mnogo ređi – ključni frejmovi. Ključni frejmovi reprezentuju najbitniji deo animacije i na osnovu njih se korišćenjem pogona u kome se razvija igra vrši interpolacija frejmova koji nedostaju. Ovo je ogromna ušteda na vremenu.

Na kraju procesa vektorske animacije obično se rasterizuju u bitmap formate, a 2D bitmap slike se često nazivaju sprajtovi.

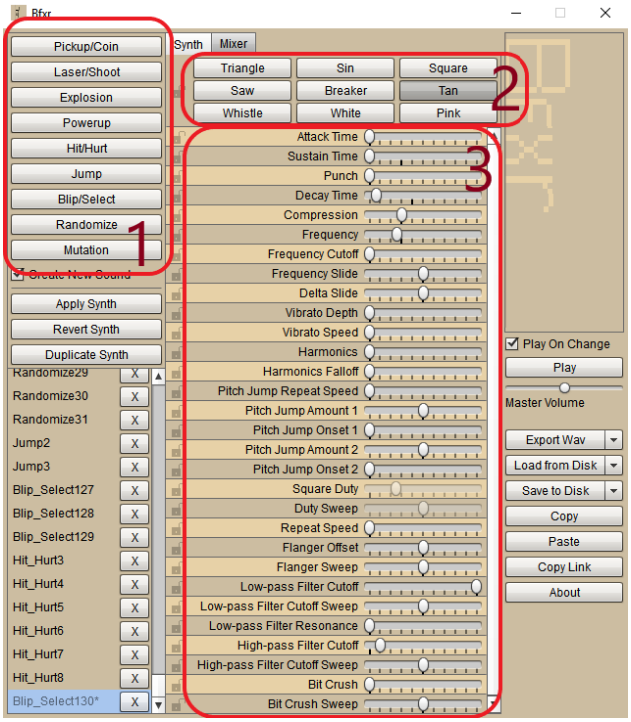
Da bi se renderovanje na ekran ubrzalo, praksa je da se velik broj sprajtova stavi u jedan fajl, a zatim taj fajl importuje u pogon u kome se razvija igra. Juniti pogon ima sistem za sečenje koji omogućava da se oni tokom razvoja koriste odvojeno, ali se svakako zajedno učitavaju u VRAM memoriju.

2.2. Zvukovni gradivni elementi

Zvučni elementi međuscena imaju neprikosnoven uticaj na atmosferu, emotivni tonalitet i osećaj uzbuđenja koji se potencijalno može dočarati korisniku. Ovaj nevidljivi uticaj postiže se pažljivim izborom elemenata muzičke podloge i zvučnih efekata koji poseduju međusobnu harmoniju tako da ni jedan ne ostavlja utisak kao da nije deo celine. Jedan od izazova u radu na ovom projektu bio je pronalazak efekata koji ispunjavaju ove kriterijume, dok sa druge strane imaju autore koji njihovo korišćenje nisu ograničili veoma restriktivnim licencama. U ovom cilju, korišćeni su – osim za dijalog – sajtovi koji ohrabruju mlade autore i ljude koji prave video igre da podele svoj rad sa drugima kao što su opengameart.org i freesound.org.

Razgovori u međuscenama su predstavljeni kroz progresivni prikaz slova na ekranu koje prate sintetički zvuci razvijeni uz pomoć programa Bfxr. Bfxr omogućava izbor velike količine parametara pri generisanju zvuka, kao što su frekvencija, oblik zvučnog talasa, njegova delta, dužina trajanja amplitude i mnogi drugi. Na slici 3 možemo videti tri izdvojene celine sa opcijama programa. U celini **1** su prečice koje omogućavaju generisanje nasumičnih zvukova koji podsećaju na zvuk raznih efekata kao što su eksplozije, udarci i slično. Zanimljivo je da je ovo odlična početna tačka jer se klikćući na neku od 9 opcija može doći do zvuka koji je blizu onoga što je željeno. Zatim, daljim korigovanjem oblika zvučnog talasa pod **2** i lepeze opcija pod **3** dolazi se do finalnog zvuka.

Izbor pod **1** u ovom slučaju uvek je bio *Blip/Select* jer su za ovu namenu bili potrebni (zbog povezanosti izgovora jednog znaka i svakog ponavljajućeg zvučnog efekta) kratki zvuci uglavnom niske frekvencije – u zavisnosti od toga koji lik govori. Visoka frekvencija može biti neprijatna kod ponavljajućih zvukova. Zanimljivo je napomenuti da svaki dodatan klik na *Blip/Select* daje sličan, ali donekle nasumičan ton, što znači da je preporučljivo kliknuti nekoliko desetina puta dok se ne dobije ton koji je najbliži onome što se traži.



Slika 3. *Opcije programa Bfxr*

Što se tiče parametara pod brojem **2**, korišćene su opcije *Whistle* i *Breaker*. *Whistle* je klasičan sinusoidni talas, sa dodatkom još jednog sinusoidnog talasa veoma niske amplitude koji ga prati na 20x većoj frekvenciji. *Breaker* je talas baziran na kvadratnoj funkciji koji je u suštini aproksimacija trougaonog talasa. Poredeći ova dva zvuka možemo uočiti da je *Breaker* homogeniji i kraći, dok je *Whistle* piskaviji i rezonantniji.

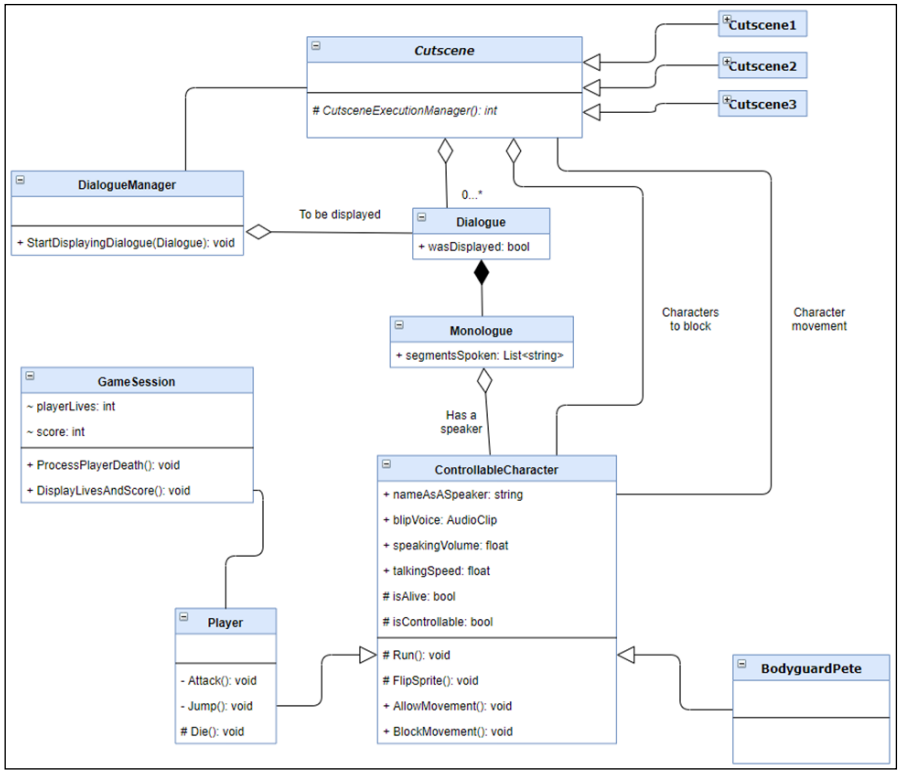
Veća izmena ostalih parametara pod **3** uglavnom nije bila potrebna jer su podrazumevane vrednosti od *Blip/Select* dale odlične rezultate. Frekvencija je naravno korigovana, a *Attack Time* i *Sustain Time* su ostali na niskim vrednostima zbog kratkog vremena trajanja koje je bitno za ovaj zvuk.

3. PROGRAMSKO REŠENJE

Elementi programskog rešenja imaju 3 prirodne celine: kontrolabilni likovi, dijalog i međuscene. Na slici 4, može se videti dijagram klasa celokupnog rešenja, na kome je očevidna povezanost između pomenutih celina – klasa *Cutscene* oslanja se na funkcionalnosti klasa *Dialogue*, *DialogueManager* i *ControllableCharacter* da bi međuscene u igri imale u sebi likove koji vode dijalog, imaju sposobnosti i kretanja i animacije koja to kretanje prati.

3.1. Dijalozi i njihov menadžer

Mnoge međuscene imaju u sebi animirane dijaloge. Svaki dijalog je definisan kao serija neprekidnih monologa (monolog može biti i samo jedna rečenica) koje vode različiti govornici. Svaki monolog se u memoriji čuva kao serija stringova koji predstavljaju rečenice koje on sadrži.



Slika 4. *Dijagram klasa programskog rešenja*

Pošto ima referencu na kontrolabilnog lika koji drži monolog, pristup klasi *Monologue* omogućava nam da odredimo i brzinu kojom govori taj lik, zvuk koji pravi, ime koje treba da se prikaže na ekranu i ostale informacije potrebne za animiranje dijalog oblačića. Osim serije monologa, klasa *Dialogue* ima i *bool* promenjivu *wasDisplayed* koja je namenjena da omogući proveru da li je dijalog prikazan. Ova promenjiva je namenjena za sinhronizaciju više dijaloga unutar međuscena, jer omogućava slanje upita instanci klase *Dialogue* koja daje odgovor na pitanje da li ju je u celosti prikazao menadžer dijaloga nakon što mu je prosleđena.

Klasa *DialogueManager* zadužena je za animaciju dijaloga na ekranu. Njene glavne metode su *StartDisplayingDialogue* i *DisplayNextSentence*.

StartDisplayingDialogue je javna metoda koja se može pozvati iz bilo koje tačke projekta, i kroz nju se pristupa svim funkcionalnostima dijalog menadžera. Ova funkcija kada dobije objekat dijaloga koji treba da prikaže izvršava animiranje otvaranja oblačića za prikaz dijaloga postavljanjem vrednosti parametra *IsOpen* u komponenti animatora na istinitu. Referenca na dijalog se čuva, a zatim se prikazuje prva rečenica prvog monologa koji on sadrži pokretanjem funkcije *DisplayNextSentence*. Pritiskom na dugme *space*, ova funkcija se pokreće ponovo i ispisuje narednu rečenicu. Kada više nema rečenica za ispis, *IsOpen* se postavlja na neistinitu vrednost, čime se animira zatvaranje oblačića za dijalog.

3.2. Međuscene

Međuscene su realizovane korišćenjem apstraktne klase *Cutscene* koja je neka vrsta šeme međuscene i konkretnih realizacija pojedinačnih međuscena koje je nasleđuju (označene su kao *Cutscene1*, *Cutscene2* i *Cutscene3* na dijagramu klasa). Klasa *Cutscene* funkcijom *OnTriggerEnter2D* proverava da li je došlo do kontakta sa objektom koji ima tag **Player**:

```
protected virtual void OnTriggerEnter2D(Collider2D collision)
{
    if (!cutsceneTriggeredAlready)
    {
        if (collision.gameObject.tag == "Player")
        {
            for(int i=0; i<charactersToBlock.Length; i++)
            {
                charactersToBlock[i].BlockMovement();
            }
            cutsceneTriggeredAlready = true;
        }
    }
}
```

Ukoliko jeste, funkcijom *BlockMovement* se blokiraju likovi iz liste kontrolabilnih likova koja se unosi iz Inspektor prozora. Na kraju, promenjiva *cutsceneTriggeredAlready* se postavlja na istinitu vrednost. Ovim se aktivira petlja u *Update* funkciji koja se poziva svaki frejm igre:

```
// Update is called once per frame
protected void Update () {
    if (cutsceneTriggeredAlready && !cutsceneCompleted)
    {
        if (CutsceneExecutionManager() == 1)
        {
            if (delay <= 0)//very small delay to avoid space from
            {           //skipping dialogue to also trigger jumping
                cutsceneCompleted = true;
                UnblockObjects();
            }
            else
            {
                delay -= Time.deltaTime;
            }
        }
    }
}

//this function should return 1 when it's done
protected abstract int CutsceneExecutionManager();
```

Ona poziva funkciju *CutsceneExecutionManager* sve dok ona ne vrati vrednost 1, a onda završava sa radom i vraća korisniku kontrolu nad nizom likova koji su prethodno blokirani. Ovo se ne čini odmah, već sa minimalnom pauzom. Razlog za postojanje pauze što je isto dugme u igri (*space*) određeno kao dugme za skok, i dugme za preskakanje trenutne rečenice dijaloga. Ako ne bi bilo pauze posle kraja dijaloga i ako bi do kraja korisnik došao preskakanjem poslednje rečenice – na kraju dijaloga bi kontrolabilni likovi izvršili akciju skok iako to ne bi bilo predviđeno ponašanje sa aspekta onoga što korisnik očekuje. Funkcija *CutsceneExecutionManager* je definisana kao apstraktna funkcija bez parametara koja vraća brojčanu vrednost. Svaka klasa koja nasleđuje klasu *Cutscene* dakle mora da sadrži implementaciju ove funkcije. Ona je namenjena da definiše sve automatizovane radnje na ekranu, kretanja igrača, dijalog i ostala ponašanja koja definišu tu jedinstvenu međuscenu. Na sledećem primeru se može videti jedna takva implementacija:

```
protected override int CutsceneExecutionManager()
{
    //display first dialogue
    if (displayDialogueOrder)...
    //lay down
    if (dialogues[0].wasDisplayed && !laidDown)...
    //dim the screen
    if (laidDown && !screenDimmingComplete)...
    //undim the screen
    if (screenDimmingComplete && !screenUndimmmingComplete)...
    //get up
    if (screenUndimmmingComplete && !gettingUpComplete)...
    //display 2nd dialogue
    if (gettingUpComplete && secondDialogueDisplayOrder)...
    //move blackopsguy and display third dialogue
    if(dialogues[1].wasDisplayed && !blackOpsGuyFinished)...
    if (dialogues[2].wasDisplayed)
    {
        return 1;
    }
    else
    {
        return 0;
    }
}
```

U međusceni u kojoj se implementira ova funkcija postoji veliki broj akcija, koje se izvršavaju hronološki zahvaljujući mnogobrojnim *if* naredbama. Npr. prva naredba se izvršava samo ako se dobije naređenje za ispis dijaloga, druga ako je dijalog ispisan izvršava animaciju leganja na zemlju, treća ako je lik legao na zemlju zamračuje ekran itd. Svaka naredba se izvršava samo ako je prethodna zadovoljena. Na kraju, ako se poslednji dijalog uspešno ispiše, vraćanjem vrednosti 1 se završava međuscena. Na ovaj način ova funkcija implementacijom samo jedne funkcije ima apsolutnu kontrolu nad događajima koji je čine i njihovim redosledom izvršavanja.

Jedna od osnovnih akcija u međuscenama – automatizovano kretanje likova – realizovano je u ovom projektu na isti sličan kao i kretanje uz pomoć tastature ili drugih ulaznih uređaja. Jedina razlika kod ovakvog kretanja je što brzina nije zavisna od ulaznih uređaja već je fiksna, a momenat (ili lokacija) početka i kraja kretanja su već unapred poznati. Momenat početka je u ovom slučaju utvrđen korišćenjem upravo pomenutih *if* petlji u funkciji *CutsceneExecutionManager* – tj. on zavisi od akcija koje prethode kretanju lika. Da bi se odredio momenat kraja, korišćen je poseban Juniti objekat sa skriptom i kolajderom koji je povezan sa njom. Kolajderi su Juniti komponente koje detektuju sudare objekata i omogućavaju programsko reagovanje na njih. Kada se lik koji se automatski kreće susretne sa kolajderom, skripta ovog objekta menja jedan od parametara skripte međuscene *Cutscene 2* i sa ovom izmenom dolazi do stopiranja kretanja lika. Ovo je veoma pogodno rešenje ukoliko je potrebno izvršiti promene jer se iz Juniti editora uvek na veoma lak način može promeniti lokacija kolajdera.

4. ZAKLJUČAK

U ovom radu su definisane međuscene kao sastavni deo igara i predstavljeni su njihovi gradivni elementi kao i načini njihovog razvoja i izbora. Nakon toga predstavljene su tehnike programskog razvoja specifične za Juniti okruženje, i na kraju, programsko rešenje za razvoj međuscena.

Jedan od problema koji se ispoljio tokom radu je relativno spora kreacija novih međuscena zbog potrebe za „ručnom“ sinhronizacijom elemenata svake međuscene jer su razvijene sa kompletnim oslanjanjem na skripte. Ovaj problem bi mogao da se reši korišćenjem *GameObjectRecorder* klase iz *UnityEditor* biblioteke. Ona se vezuje za Juniti objekte i snima vrednosti u vezi sa njihovom promenom položaja u sceni. Upotrebom ove klase bilo bi moguće snimiti pokrete likova koji bi se kasnije automatski pomerali na isti način. Ovo bi drastično ubrzalo rad na međuscenama.

Dalji pravac razvoja mogao bi biti razvoj međuscena sa grananjem, koje bi omogućile da se priča igre račva na nekoliko mesta. Ovako bi se igraču vratio stepen kontrole, čiji gubitak je jedan od čestih kritika u igrama koje imaju velik broj međuscena.

5. LITERATURA

[1] https://boingboing.net/2018/07/17/the-most-popular-engines-for-i.html (pristupljeno u oktobru 2019.)

[2] S. Lavelle, „Bfxr – Make Sound Effects for Your Games“, https://www.bfxr.net/, Bfxr – program za generisanje zvučnih efekata.

Kratka biografija:

Stevan Zivlak rođen je u Novom Sadu 1990. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – odbranio je 2019.god.
kontakt: szivlak@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE37Baštovanovic

GENERATOR STATIČKIH I DINAMIČKIH WEB APLIKACIJA

STATIC AND DINAMIC WEB APPLICATION GENERATOR

Nikola Baštovanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu predstavljen je jezik i generator programskog koda za statičke i dinamičke web aplikacije. Generator kao ulaz koristi model pisan na internom jeziku specifičnom za domen (DSL-u). DSL je implementiran u okviru XML-a. Generisanje koda se vrši uz pomoć T4 tekstualnog obrađivača šablona, kao i korišćenjem parsera jezika koji se koristi na ciljanoj platformi.*
Ključne reči: *Generator koda, C#, .Net, ASP.Net, Sql Server, T4 obrađivač šablona*
Abstract – *This paper presents the language and static and dinamic web application code generator. Code generator uses model written in internal Domain Specific Language (DSL) as an input. Code generation is performed using T4 template engine and also by using parsers for target platform languages.*
Keywords: *Code generator, C#, .Net, ASP.Net, Sql Server, T4 template engine*

1. UVOD

Prilikom razvoja web aplikacija, oslanjamo se na slične ili čak identične koncepte. Kreiranje html stranica koje se jedna od druge razlikuju u fontu, formatiranju, boji teksta i slično, a sve specificirano u css klasama. Imajući sve ovo u vidu, zašto bi iznova i iznova pisali kod koji je već u velikoj meri napisan? Odgovor na ovo pitanje je tema ovog rada. Razvoj generatora koji će pojednostaviti razvoj ovakvih aplikacija.

Osnovni zadatak generatora programskog koda jeste da generiše kod koji je ispravan, dakle kod koji može da se pokrene bez prikazivanja grešaka. Ono što je takođe veoma bitno kod generatora programskog koda je da kod koji je generisan bude što je moguće efikasniji.

U okviru ovog rada biće prikazan generator programskog koda koji kao ulaz koristi model pisan na internom DSL-u (Domain-Specific Language). DSL je implementiran u okviru XML-a (Extensible Markup Language).

Na osnovu ulaznog modela generator će kreirati: skript za kreiranje šeme baze podataka, osnovne operacije nad bazom podataka, kao i predefinisani izgled samih stranica koje korisnik može naknadno menjati u skladu sa svojim potrebama, željama i zahtevima.
S obzirom da generator, pored statičkih, generiše i dinamičke ASP.Net aplikacije, generator će na osnovu ulaznog modela generisati i ASP.Net klase koje omogućavaju pravilan rad generisane aplikacije.

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

2 PROJEKTOVANJE I IMPLEMENTACIJA TEHNIČKOG REŠENJA

U okviru ovog poglavlja je opisan jezik specifičan za domen, neke od njegovih prednosti i mana, implementacija internog jezika specifičnog za domen u XML formatu. Takođe, dat je prikaz dijagrama aktivnosti korišćenja sa detaljnim objašnjenjem, Kratak opis i primeri upotrebe T4 tekstualnog obrađivača šablona, osnove Entity Framework-a i Owin-a.

2.1 Jezik specifičan za domen

Jezik specifičan za domen (Domain-specific language - DSL) je programski jezik koji je specifičan za određeni, usko definisan domen primene. Za razliku od jezika opšte namene (General Purpose Language - GPL) koji se široko primenjuju za različite namene i nedostaju mu specijalizovane funkcije za određenu oblast, nude povećanje ekspresivnosti koje se postiže kroz upotrebu koncepata i notacija prilagođenih domenu problema i domenskih eksperata [2].
Jezici specifični za domen se mogu podeliti prema vrsti jezika [1]:

1. - Domenski specifični jezici za obeležavanje;
2. - Domenski specifični jezici za modelovanje i
3. - Domenski specifični programski jezici.

Kada je reč o upotrebi jezika specifičnog za domen, onda treba napomenuti da kreiranje ovakvog jezika ima smisla samo onda kada takav, domenski specifičan jezik, omogućava da se određena vrsta problema ili rešenja izrazi jasnije nego što bi to dozvoljavao neki od postojećih jezika, ali i da se tip problema koji se tim načinom rešava pojavljuje dovoljno često kako bi ovakav metod bio opravdan i isplativ.

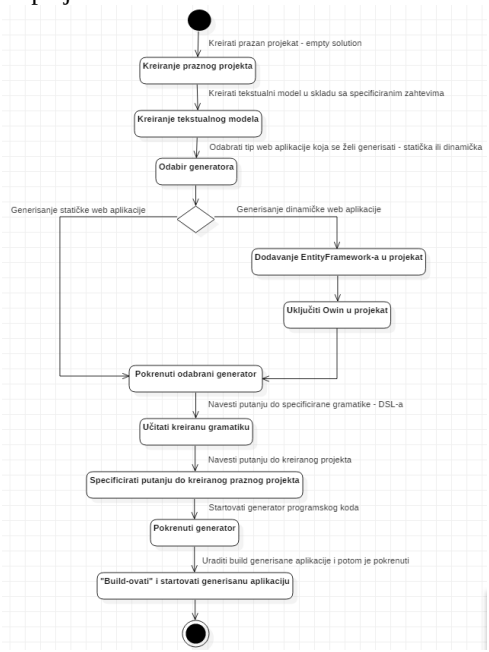
2.2 Dijagram aktivnosti

Na slici 1 je dat dijagram aktivnosti korišćenja sistema koji se razvija u ovom radu.

2.3 T4 Tempalte

U ovom projektu korišćen je C# T4 obrađivač šablona. Šabloni pisani na jeziku datog obrađivača predstavljaju kombinaciju tekstualnih blokova i kontrolne logike pri kreiranju tekstualnih fajlova koji mogu biti različitih ekstenzija. Neki od tipova ekstenzija koji su korišćeni u projektu su: .sql koji predstavlja šablon za generisanje sql skripta koji se pušta nad bazom podataka, .cs što predstavlja šablon za generisanje klasa u C# programskom kodu, .aspx za generisanje ekstenzija

ASP.Net klase, i druge. Blokovi koji sadrže logiku sadrže C# programski kod sa svim svojim pogodnostima koje C# programski jezik pruža [5].
Kada je reč o T4 templejtu, razlikuju se dva osnovna tipa ovog templejta:



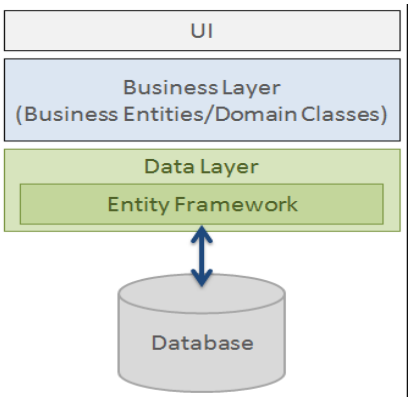
Slika 1: *Dijagram aktivnosti generatora programskog koda*

- RunTime - koji se izvršavaju, kako im i naziv kaže, u vreme izvršavanja aplikacije i generišu kod koji se direktno koristi u vreme izvršavanja aplikacije.
- - DesignTime - generišu kod koji se kasnije koristi u aplikaciji.

2.4 Entity Framework

Entity Framework je delo Microsoft korporacije koji u velikoj meri olakšava razvoj web aplikacija. Kreiranje konekcija, vođenje računa o krajnim rezultatima, upisu u bazu, izmeni podataka i brisanju istih iz pojedinih tabela, o svemu tome sada brine ovaj okvir.

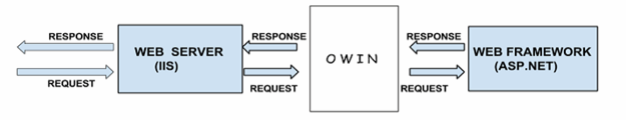
Entity Framework je objektno orijetisan razvojni okvir otvorenog koda koji programerima omogućava rad sa objektima koji imaju vezu sa podacima u bazi podataka bez potrebe fokusiranja na tabele i kolone gde se ti podaci nalaze. Omogućava kreiranje aplikacija sa mnogo manje ručno pisanog programskog koda. Slika 2 prikazuje poziciju razvojnog okvira u aplikaciji [6]:



Slika 2: *Pozicija Entity Framework-a u web aplikaciji* [6]

2.5 Owin

Owin predstavlja standardni interfejs između .Net web servisa i web aplikacije. Glavni cilj Owin-a je da razdvoji server i aplikaciju, podstakne razvoj jednostavnih modula za razvoj .Net web aplikacije i slično. OAuth autorizacija omogućava aplikacijama ograničen pristup Http servisima.
Slika 3 prikazuje cilj Owin-a, a to je da razdvoji server od aplikacije [7].

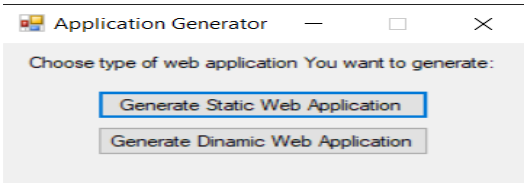


Slika 3: Owin razdvaja server i aplikaciju [7]

3. PRIKAZ IMPLEMENTIRANOG REŠENJA

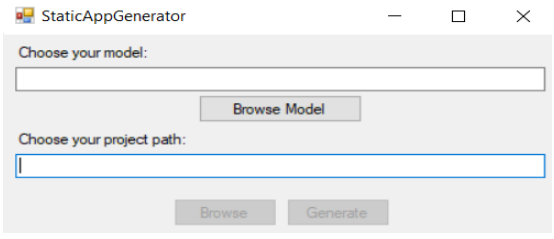
U okviru ovog poglavlja biće detaljno opisan proces generisanja programskog koda za statičke web aplikacije a zatim i za dinamičke web aplikacije. Osnovna razlika statičkog i dinamičkog web sajta je u tome ko ima prava da menja sadržaj samog sajta.

Pod statičkom web aplikacijom se podrazumeva aplikacija čiji sadržaj može da menja i nadograđuje samo stručno lice, kao što je web dizajner ili programer, dok se pod dinamičkom web aplikacijom podrazumeva web aplikacija čiji sadržaj može da menja svaki korisnik računara.
Pokretanjem implementiranog generatora prikazuje se korisnički interfejs kao na slici 4 ispod.



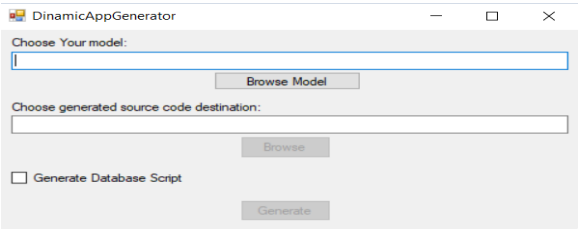
Slika 4: Korisnički interfejs generatora web aplikacija

Kao što se to može videti na slici 4, korisnik može odabrati generisanje statičke ili dinamičke web aplikacije. Klikom na dugme *Generate Static Web Application* otvara se prozor kao što je to prikazano na slici 5.



Slika 5: Generator za generisanje statičke web aplikacije

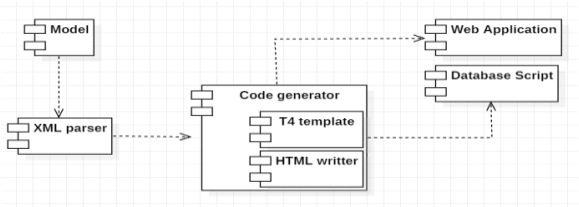
Klikom na dugme *Generate Dinamic Web Application* otvara se prozor kao što je to prikazano na slici 6.



Slika 6: *Generator za generisanje dinamičke web aplikacije*

3.1 Programski kod generatora i generisani programski kod

Najbolji način prikaza funkcionisanja generatora je upoznavanje sa njegovom arhitekturom. Slika 7 predstavlja arhitekturu generatora web aplikacija.



Slika 7: *Arhitektura generatora programskog koda*

Na slici 7 se može videti koje su to komponente koje predstavljaju ulaz u generator, a koje izlaz. Nakon što je generatoru prosleđen model, on prolazi kroz XML parser, budući da se radi o modelu u XML formatu. Kada je isparsiran, prolazi kroz T4 tekstualni templejt i kroz HTML writter kako bi se izvršilo generisanje fajlova potrebnih za pravilan rad web aplikacije. Jedan od izlaznih fajlova može biti i .sql fajl koji predstavlja skript za kreiranje baze podataka.

Da bi opis rada generatora bio potpun sledi detaljnije objašnjenje generisanja programskog koda. Najpre će biti prikazan programski kod generatora a zatim generisani programski kod dinamičke web aplikacije. Generator je implementiran na način da najpre generiše HTML stranice. Za generisanje HTML stranica korišćena je klasa *HtmlTextWriter* koja umnogome olakšava generisanje krajnjeg HTML koda korišćenjem C# programskog jezika. Slika 8 daje prikaz koda generatora.

```
using (HtmlTextWriter writer = new HtmlTextWriter(stringWriter))
{
    writer.RenderBeginTag(HtmlTextWriterTag.Html);

    writer.AddAttribute("runat", "server");
    writer.RenderBeginTag(HtmlTextWriterTag.Head);

    writer.RenderBeginTag(HtmlTextWriterTag.Title);
    writer.Write(myAppName);
    writer.RenderEndTag(); //End title tag
    writer.WriteLine();

    writer.WriteBeginTag("link");
    writer.WriteAttribute("rel", "stylesheet");
    writer.WriteAttribute("href", "Styles/StyleSheet.css");
    writer.WriteAttribute("type", "text/css");
    writer.Write(HtmlTextWriter.SelfClosingTagEnd); //End link tag
    writer.WriteLine();

    writer.AddAttribute("ID", "head");
    writer.AddAttribute("runat", "server");
    writer.RenderBeginTag("asp:ContentPlaceHolder");
    writer.RenderEndTag(); //End asp:ContentPlaceHolder tag
    writer.WriteLine();

    writer.RenderEndTag(); //End head tag
    writer.WriteLine();

    writer.RenderBeginTag(HtmlTextWriterTag.Body);
```

Slika 8: *Prikaz programskog koda generatora za generisanje HTML stranice korišćenjem HtmlTextWriter*-a

Pokretanjem ovog koda, generator će izgenerisati HTML programski kod koji je prikazan na slici 9.

```
<html>
    <head runat="server">
        <title>
            DinamicTest
        </title>
        <link rel="stylesheet" href="Styles/StyleSheet.css" type="text/css" />
        <asp:ContentPlaceHolder ID="head" runat="server">
        </asp:ContentPlaceHolder>

    </head>
    <body>
```

Slika 9: *HTML programski kod izgenerisan pokretanjem generatora*

Ovim je dat kratak primer i objašnjenje generisanja koda koji se tiče izgleda stranica. Sledi prikaz i objašnjenje koda generatora koji kreira C# klase. Za ovo je korišćen T4 templejt.

Korišćenje ovog templejta biće objašnjeno kroz generisanje jedne od metoda koja se koristi pri komuniciranju sa bazom podataka. Na slici 10 je dat prikaz tog koda u T4 templejtu.

```
  foreach (XmlNode xmlNodeTableName in doc.GetElementsByTagName("name"))
  {
      string modelName = xmlNodeTableName.InnerText;
#>
using <#= myAppName #>;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace <#= myAppName #>.Models
{
    public partial class <#= modelName #>Model
    {
        public string Insert<#= modelName #>(<#= modelName #> <#= modelName.ToLower() #>)
        {
            try
            {
                <#= dbName #>Entities db = new <#= dbName #>Entities();
                db.<#= modelName #>s.Add(<#= modelName.ToLower() #>);
                BeforeInsert(<#= modelName.ToLower() #>);
                db.SaveChanges();

                return <#= modelName.ToLower() #>.ID + " was succesufully inserted.";
            }
            catch (Exception e)
            {
                return "Error: " + e;
            }
        }
```

Slika 10: *T4 templejt za generisanje Insert metode*

Slika 10 daje prikaz koda generatora koji kreira Insert metodu koja komunicira sa bazom podataka i vrši upis novih podataka u bazu. Na samom početku slike 10 se vidi *foreach* petlja koja omogućava kreiranje Insert metoda za svaki model iz gramatike. Kod koji se nalazi posle odrednice #> će kao takav da se nađe u izgenerisanoj metodi. Svaka promenljiva koja se nađe u okviru tagova <#= i #> biće zamenjena vrednošću te promenljive u metodi koja se nađe u generisanoj web aplikaciji. Slika 11 potvrđuje to dajući prikaz generisane Insert metode.

```
using DinamicTest;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;


namespace GeneratedDinamicWebSite.Models
{
    0 references
    public partial class ProductModel
    {
        0 references
        public string InsertProduct(Product product)
        {
            try
            {
                GarageEntities db = new GarageEntities();
                db.Products.Add(product);
                BeforeInsert(product);
                db.SaveChanges();

                return product.ID + " was succesufully inserted.";
            }
            catch (Exception e)
            {
                return "Error: " + e;
            }
        }
```

Slika 11: *Generisan programski kod*

Kao što se može videti, generisana je parcijalna klasa. To omogućava korisniku da nakon generisanja programskog koda doda i svoj, ručno pisan kod. Metoda *BeforeInsert*, ovde pozvana, parcijalna je metoda.

Ukoliko korisnik naknadno ne kreira ovu metodu, kompajler će ignorisati taj deo koda i programski kod će biti kompajliran bez problema.

Ukoliko, pak, korisnik kreira tu parcijalnu metodu, kompajler će uzeti u obzir novu implementaciju *BeforeInsert* metode. Nakon ponovnog pokretanja generatora, ručno pisani kod koji je korisnik naknadno pisao neće biti "pregažen" generisanim kodom.

4. ZAKLJUČAK

U ovom radu prikazano je generisanje statičkih i dinamičkih web aplikacija. Kreiranjem ovakvog generatora posao razvoja aplikacija umnogome je olakšan i ubrzan.

Na krajnjem korisniku je da personalizuje generisanu web aplikaciju u skladu sa svojim potrebama i željama ili da je koristi onako kako je generator izgenerisao. Generator daje potpuno funkcionalan programski kod koji je direktno upotrebljiv bez naknadnih intervencija.

Za implementaciju je korišćen interni domenski specifičan jezik razvijen u okviru XML formata koji je detaljno objašnjen. Kroz primer koji je predočen u ovom radu, moglo se videti kako naizgled jednostavan DSL može dovesti do implementacije jedne potpuno ispravne i kompleksne web aplikacije.

Aplikacija koja je izgenerisana na način ovde prikazan omogućava korisniku da kombinuje generisani programski kod i ručno pisani kod ukoliko postoji potreba za daljim izmenama aplikacije.

5. LITERATURA

[1] Wikipedia, Domenski specifični jezici, https://en.wikipedia.org/wiki/Domain-specific_language

[2] Igor Dejanović, Kurs iz predmeta Jezici specifični za domen, http://www.igordejanovic.net/courses/jsd/uvod.html#7

[3] Metodologija razvoja softvera, http://www.tfzr.uns.ac.rs/Content/files/0/MRSLab03%20-%20PDF%20-%20R1.pdf

[4] Dijagram aktivnosti, http://www.vps.ns.ac.rs/Materijal/mat3542.pdf

[5] Microsoft Documents, Code Generation And T4 Templates, https://docs.microsoft.com/en-us/visualstudio/modeling/code-generation-and-t4-text-templates?view=vs-2019

[6] Entity Framework Tutorial, https://www.entityframeworktutorial.net/what-is-entityframework.aspx

[7] Owin, http://owin.org/ Jignesh Trivedi, Token Based Authentication Using ASP.Net Web API, OWIN and Identity With Entity Framework, https://www.c-sharpcorner.com/UploadFile/ff2f08/token-based-authentication-using-Asp-Net-web-api-owin-and-i/

6. BIOGRAFIJA

Nikola Baštovanović rođen je 26. januara 1993. godine u Bajinoj Bašti, Republika Srbija. Osnovnu školu „Rajak Pavićević" i gimnaziju „Josif Pančić" završio je u Bajinoj Bašti sa odličnim uspehom. Fakultet tehničkih nauka u Čačku, smer Informacione tehnologije modul Softversko inženjerstvo upisao je 2012. godine. Akademske studije završio je 2016. godine sa prosečnom ocenom 9.56. Iste godine upisuje master akademske studije na Fakultetu tehničkih nauka u Novom Sadu, smer Softversko inženjerstvo i informacione tehnologije.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.41
DOI: https://doi.org/10.24867/07BE38Marojevic

DSL I GENERATOR KODA ZA PODRŠKU DIGITALIZACIJE NAFTNIH POLJA

DSL AND CODE GENERATOR FOR DIGITALIZATION OF OIL FIELDS SUPPORT

Ana Marojević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – SOFTVERSKO INŽENJERSTVO I INFORMACIONE TEHNOLOGIJE

Kratak sadržaj – *U ovom radu predstavljeni su domen specifičan jezik za podršku digitalizacije naftnih polja koji obuhvata definisanje podataka, podršku za višejezički korisnički interfejs, sistem za konverziju veličina u različite jedinice i generator osnove projekta u skladu sa takvom definicijom. Jezik je implementiran uz oslonac na Microsoft DSL Tools alate i T4 template za generisanje izlaza.*

Ključne reči: *Generator koda, domen specifičan jezik, Microsoft DSL Tools*

Abstract – *This paper presents a domain-specific language for digitalization of oil fields support that includes defining data, multilingual support, unit conversion system and corresonding code generator. The language is implemented using Microsoft DSL Tools and generator uses T4 template for generating output.*

Keywords: *Code Generator, Domain Specific Language, Microsoft DSL Tools*

1. UVOD

Danas softver kontroliše mnogo aspekata našeg života. Rad programera sastoji se u mapiranju konkretnog problema na rešenje koje razrešava dati problem. Da bi zabeležili takvo rešenje, programeri koriste jezike za programiranje. Programski jezik pruža sloj između programera i postojeće hardverske i/ili softverske platforme na kojoj se program izvodi [1]. Kako bi se olakšao rad, taj međusloj može se premestiti bliže programeru, tj. može se smanjiti jaz između domena problema i domena rešenja. Programski jezici koji imaju za cilj poboljšanje mapiranja problema na rešenje poznati su kao domen specifični jezici (*DSL – Domain Specific Languages*), za razliku od jezika opšte namene (*GPL – General Purpose Languages*).

Modelovanje i definisanje jezika specifičnih za domen u poslednjoj deceniji su znatno napredovali, što je omogućilo da programeri i domenski eksperti mogu da se fokusiraju na konkretne domenske probleme koje rešavaju. Izrada poslovnog sistema podrazumeva implementaciju više stotina međusobno povezanih domenskih entiteta i bar toliko ekranskih formi.

Implementacija na klasičan način bi podrazumevala ručno kodiranje što dovodi do velikih problema ne samo u izgradnji takvog sistema, već i u održavanju i migraciji sistema na nove platforme [2].

Motivacija za ovaj master rad javila se kao odgovor na realan problem jednog softverskog proizvoda u oblasti digitalizacije naftnih polja sa posebnim akcentom na proračun preostalih naftnih rezervi. Rukovodioci projekta su naftni inženjeri koji, iako informatički obrazovani, nemaju mnogo dodira sa programiranjem. S obzirom da su podaci usko vezani za domen rudarstva i naftne industrije, a programeri nemaju dovoljno znanja iz ove oblasti, bilo je neophodno osmisliti rešenje da naftni inženjeri budu ti koji će kreirati opis podataka na dovoljno jednostavan način, a programeri dobiti te podatke u obliku prilagođenom njihovim potrebama. Ideja je da se izbegne definisanje podataka na papiru ili elektronskom obliku od strane inženjera a zatim da programeri isto to pretvaraju u programski kod. Pored toga, već je postojala konvencija za definisanje podataka iz prethodnog softverskog rešenja u vidu tri XML (*eXtensible Markup Language*) dokumenta koju su inženjeri hteli da zadrže, te je bilo potrebno prilagoditi se takvoj strukturi. Aplikacija je tehničke prirode i koristi ogroman broj podataka koji se u naftnoj industriji izražavaju u nekoliko različitih sistema mernih jedinica. Te podatke je potrebno definisati prilikom kreiranja modela, što je bio dodatan motiv za nastanak generatora kako bi se izbegle greške prilikom ručnog pisanja koda. Zamisao je da inženjeri formulišu jedinstveni opis podataka koji se skladišti na jednom mestu, a da se implementira generator koji će kreirati i dobavljati podatke iz baze u obliku prilagođenom aplikaciji koja koristi tu bazu podataka.

2. DEFINICIJA POJMOVA

2.1. Model

Model predstavlja opis, ili specifikaciju sistema i njegovog okruženja kreiranu za određenu namenu. Najčešće je model predstavljen kao kombinacija crteža i teksta. Tekst može biti zadat jezikom za modelovanje ili prirodnim jezikom [3].

Model je apstrakcija nečega što u stvarnosti postoji. Nikada nije preslikana slika realnog sveta, uvek se razlikuje od onoga što modeluje. Često su detalji izostavljeni u modelu. Model nikada neće dati potpuno iste odgovore kao modelovani sistem ali se može očekivati da te razlike budu u projektovanim granicama.

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

2.2. Meta-model

Meta-model je model koji definiše apstraktnu sintaksu jezika koji se koristi da opiše model. Apstraktna sintaksa određuje pravila validnosti jezičkih iskaza sa stanovišta njegove strukture bez razmatranja konkretne reprezentacije iskaza (konkretne sintakse) [4]. Svaki meta-model je takođe model i sadrži koncepte domena, njihove međusobne veze i ograničenja.

2.3. Transformacija

Transformacija modela je proces pretvaranja jednog modela u drugi model istog sistema.

Kada je reč o arhitekturi vođenoj modelima, model nezavisan od platforme i ostale informacije se kombinuju transformacijom kako bi se dobio model specifičan za platformu. Zatim se drugom transformacijom model specifičan za platformu prevodi u kod. Ove transformacije su ključne za razvojni proces arhitekture vođene modelima.

Izlaz transformacije se definiše definicijom transformacije. Ona se sastoji od kolekcije pravila transformacije, koje nedvosmisleno specificiraju način na koji se jedan deo modela koristi za kreiranje dela drugog modela.

2.4. Generator koda

Generator koda je alat ili resurs koji generiše određeni kod. Postoji nekoliko pristupa generisanju koda. Prvi je naivan pristup koji se izvodi kombinovanjem fragmenata koda upotrebom komandi print oblika. Kada je većina generisanog koda fiksna i samo određeni delovi koda zavise od modela, tada je preporučljivo koristiti obrađivače šablona (*template engines*) jer oni omogućavaju umetanje varijabilnih delova u ostatak koda. Fiksni delovi generisanog koda su definisani bez izmena, a varijabilni delovi su definisani upotrebom posebnih iskaza šablona. Obrađivač šablona kasnije te iskaze interpretira.

Osnovni razlozi za generisanje koda su [5]:

- Produktivnost – jednom napisan generator koda može se iskoristiti koliko god puta je potrebno. Davanje specifičnih ulaza i pozivanje generatora znatno je brže od pisanja koda ručno, stoga omogućava uštedu vremena
- Jednostavnost – generisanje koda se vrši iz apstraktnog opisa što znači da je izvor poverenje taj opis, a ne kod. Takav opis je obično lakše analizirati i proveriti u poređenju sa celim generisanim kodom
- Prenosivost – isti apstraktni opis se može koristiti za stvaranje različitih vrsta artefakata
- Konzistentnost – generisanjem koda se uvek dobija kod koji se očekuje, naravno osim u slučaju grešaka u generatoru, pa je samim tim i kvalitet koda konzistentan

2.5. Jezici specifični za domen

Jezici opšte namene su dovoljno dobri alati za sve vrste programa, ali nisu specifični za određenu oblast. Čak i u polju opšteg programiranja postoje različiti jezici, od kojih svaki pruža različite prednosti za rešavanje određenih zadataka. Što su zadaci specifičniji, više je razloga za korišćenje jezika specifičnih za domen.

Jezik specifičan za domen je jezik koji je optimizovan za određenu vrstu problema, koji se naziva domen. Zasniva se na apstrakcijama koje su usko usklađene sa domenom za koji je jezik osmišljen.

3. IMPLEMENTACIJA REŠENJA

Sistem se sastoji iz tri samostalne komponente:

- DSL-a za formulisanje opisa podataka i kreiranje XML zapisa iz formirane definicije
- Generatora klasa entiteta iz XML definicije, klasa koje odgovaraju povratnim vrednostima procedura, sloja za komunikaciju sa bazom podatka i podrškom za dependency injection mehanizam
- Konzolne aplikacije za generisanje JSON (*JavaScript Object Notation*) fajlova za podršku za rad sa multi-jezičkim korisničkim interfejsom i sistemom za konverziju veličina u različite sisteme jedinica

3.1. DSL za opis definicije podataka

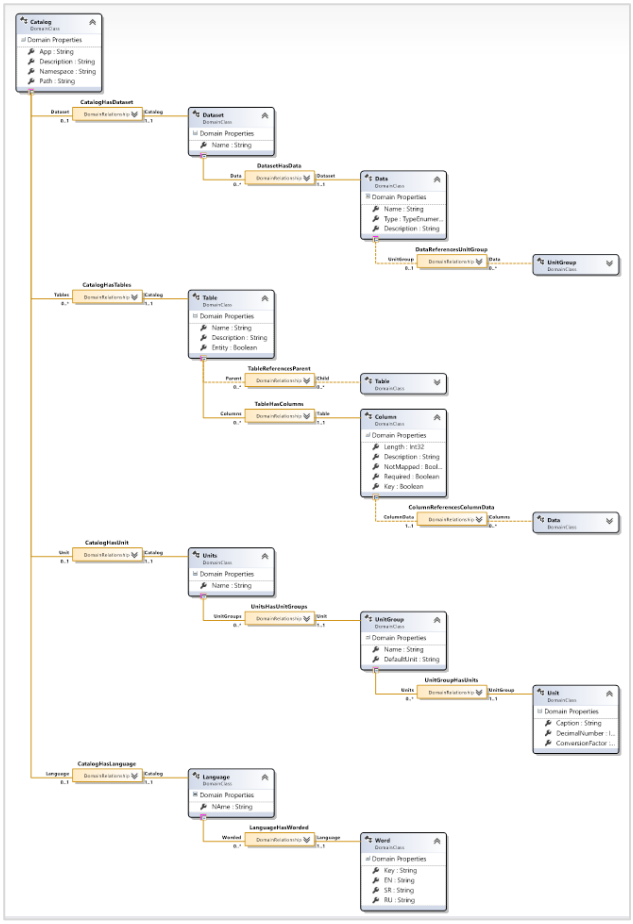
Prva komponenta sistema jeste DSL za opis podataka sistema i generisanje XML datoteka iz opisane definicije. Oslanjajući se na Microsoft DSL Tools, kreiran je meta-model i editor modela.

Definicija DSL-a obuhvata dva aspekta, izgled elemenata modela i informacije koje model nosi. Osnovni pojam u jeziku je katalog (*Catalog*). Katalog predstavlja korensku klasu i objedinjava sve ostale elemente modela. Katalog u sebi sadrži podatke vezane za projekat kao što su naziv aplikacije, njen opis, namespace koji će se koristiti za generisanje klasa i putanju do projekta. Katalog ima četiri direktna podelementa i to su definicija svih podataka (*Dataset*), tabele (*Table*), merne jedinice (*Units*) i jezici (*Language*). Meta-model prikazan je na Slici 1.

Dataset element označava skup svih podataka neophodnih za rad jednog sistema. Svaki podatak koji će se koristiti u aplikaciji mora biti definisan unutar ovog elementa, bilo da je to podatak koji se skladišti u bazu podataka ili virtuelni podatak neophodan za implementaciju nekog dela aplikacije.

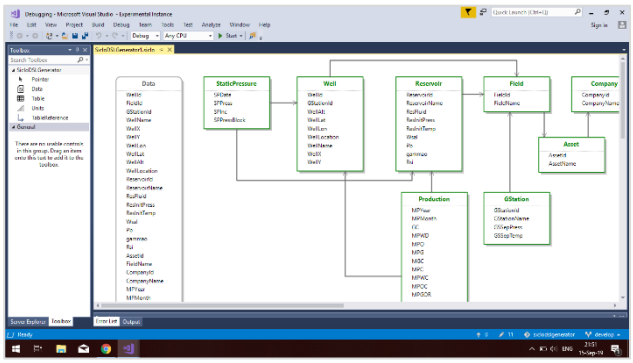
Domain klasa *Table* reprezentuje fizičku tabelu u bazi podataka. U okviru tabele moguće je definisati naziv tabele, opis i označiti da li je potrebno generisati dodatne anotacije za Entity Framework ORM (*Object Relational Mapper*) i sloj aplikacije koji komunicira sa bazom podataka. Svaka tabela se sastoji iz kolona, na nivou kojih se ograničava maksimalna dužina polja, obaveznost polja, da li se radi o virtuelnom podatku i da li je kolona deo ključa tabele. Kolona mora biti vezana sa instancom *Data* klase, koju je potrebno prethodno definisati.

Sistem za konverziju podataka u različite merne jedinice oslanja se na *Units Domain* klasu. Ona reprezentuje skup svih grupa mernih jedinica (npr. temperatura, pritisak, dužina, itd.). Svakom podatku koji predstavlja veličinu se dodeljuje grupa kojoj pripada, koja ima svoj naziv i osnovnu jedinicu u kojoj je vrednost tog podatka zapisana u bazi i na osnovu nje se izračunavaju izvedene jedinice. Svaka od grupa mernih jedinica sadrži listu jedinica (*Unit*) sa svojim nazivom, brojem decimala sa kojim se prikazuje i konverzionim faktorom koji služi za pretvaranje vrednosti podatka iz osnovne jedinice u izvedenu.



Slika 1. *Meta-model jezika za opis definicije podataka*

Language element označava rečnik aplikacije za podršku višejezičkog korisničkog interfejsa. Postoji skup predefinisanih jezika koji će se koristiti u sistemu i za svaku reč ili skup reči koji će se prikazivati na klijentskoj aplikaciji unosi se adekvatan prevod za dati jezik.



Slika 2 – *Primer kreiranja modela*

Unutar domen specifičnog jezika moguće je definisati ograničenja radi provere ispravnosti modela. Microsoft DSL Tools ima ugrađenu validaciju ispravnosti kreiranja elemenata i veza između njih u skladu sa definisanom semantikom jezika. Parcijalne klase u C#-u omogućuju definisanje iste klase na više mesta. Ova tehnika pruža mogućnost da se generisani kod odvoji od ručno pisanog koda. Dodatna ograničenja se definišu dodavanjem metoda validacije u klase domena ili klase relacija. Kada se pokrene provera, bilo od strane korisnika ili pod kontrolom programa, određene ili sve validacione metode se izvršavaju. Svaka metoda se primenjuje na svaku instancu klase, a u svakoj klasi može biti nekoliko metoda provere. Metoda validacije izveštava o svim greškama i nepravilnostima modela.

3.2. Generator koda

Generator osnove projekta iz XML definicije je Windows Forms aplikacija koja generiše sve entitete iz određene baze podataka i pruža temelj za izradu projekta. Generator koristi tri XML dokumenta na osnovu kojih generiše željene izlaze. *Catalog*, *Units* i *Dictionary* dokumenti nalaze se u bazi sačuvani kao binarni fajlovi.

Aplikacija za koju se generiše osnova, za sada i jedina, je ASP.NET veb aplikacija za digitalizaciju naftnih polja. ASP.NET Web API je radni okvir koji olakšava implementaciju HTTP servisa. Predstavlja idealnu platformu za izgradnju RESTful aplikacija u .NET Framework-u.

Sa Entity Framework-om pristup podacima se vrši pomoću modela. Model se sastoji od klasa entiteta i kontekstnog objekta koji predstavlja sesiju sa bazom podataka i omogućava kreiranje upita i čuvanje podataka. Pored svojstava klase i anotacija, neophodnih za rad Entity Framework-a, generiše se atribut „*Property*“ iznad svojstava koja se mogu izraziti u različitim mernim jedinicama.

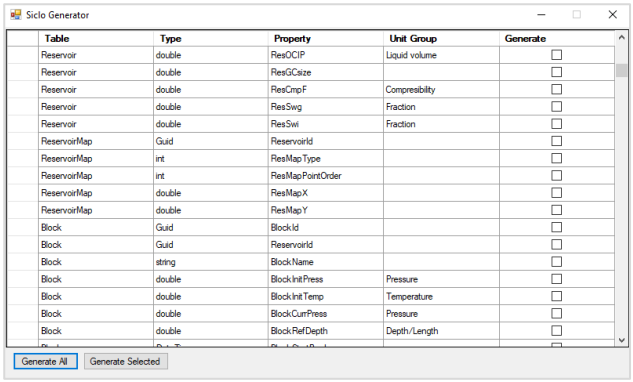
DbContext klasa je sastavni deo Entity Framework okvira. Instanca *DbContext* klase predstavlja most između entiteta i baze podataka i omogućava upravljanje konekcijom, konfigurisanje modela i relacija, olakšava upite, skladištenje, praćenje promena i mnoge druge stvari vezane za rad sa bazom podataka. Generiše se klasa koja nasleđuje *DbContext* i definiše kolekciju *DbSet*-ova koji reprezentuju kolekcije entiteta u kontekstu. *OnModelCreating()* metoda ima instancu *ModelBuilder* klase kao parametar i poziva se od strane framework-a kada se kontekst prvi put kreira kako bi se izgradio model i njegova mapiranja u memoriji.

Implementacija *Repository pattern*-a pomaže da izolujemo pristup podacima od ostatka aplikacije. Repozitorijum klasa koristi Entity Framework-ovu kontekstnu klasu podataka za obavljanje CRUD (*Create, read , update and delete*) operacija. Generator podržava generisanje ovih tipičnih metoda upravljanja podacima, a to su metode *FindAll(), Create(), Update() i Delete().*

Dependency injection (DI) je tehnika za razvijanje aplikacije sa što većim stepenom nezavisnosti. Nezavisnosti u smislu da svaki modul aplikacije treba da bude jedinstven i da ne zavisi od ostalih modula. Generatorom je omogućena osnova i olakšana upotreba *dependency injection* mehanizma, koristeći Simple Injector *open source* biblioteku za .NET.

Koristeći fiksne blokove teksta i umetanjem varijabilnih delova iz modela kreiraju se *.cs* fajlovi na izabranoj putanji. S obzirom da se u aplikaciji za koju je generator prvobitno pravljen većina podataka dobija pozivima uskladištenih SQL (*Structured Query Language*) procedura, koje ne vraćaju entitetske klase, generator je proširen delom za generisanje klasa sa proizvoljnim svojstvima, koja u ovom slučaju odgovaraju povratnim vrednostima procedura

Generatorom je omogućeno brže i efikasnije kreiranje temelja projekta i izbegavanje grešaka prilikom ručno pisanog koda, što i jeste osnovna prednost svakog generatora. Na slici 3 prikazan je izgled aplikacije za generisanje podataka.



Slika 3. *Izgled aplikacije za generisanje podataka na osnovu XML zapisa*

3.3. Aplikacija za generisanje JSON dokumenata za merne jedinice i višejezičku podršku

Generator JSON fajlova za implementaciju konverzije veličina u merne jedinice i podršku višejezičkog korisničkog interfejsa je konzolna aplikacija koja ima za cilj da izgeneriše JSON fajlove sa neophodnim podacima za prethodno navedene funkcionalnosti.

Sve veličine u aplikaciji mogu izraziti u *API* i *SI* sistemu jedinica. Pored toga, postoji mogućnost kreiranja sopstvenog sistema jedinica, odabirom željene jedinice za svaku grupu jedinica, ali takav scenario nije bilo moguće obuhvatiti generatorom.

Prvi korak pri generisanju jeste čitanje potrebnih XML dokumenata iz baze podataka. XML dokumenti se nalaze upisani kao binarni fajlovi u jednoj od tabela. Upravljanje XML dokumentom, njegovo učitavanje i pronalaženje elementa implementirana je uz oslonac na *XmlDocument* klasu .NET Framework-a. Nakon dobijanja neophodnih informacija kreiraju se JSON fajlovi koji se koriste za konverziju veličina u različite jedinice i podršku za multi-jezički korisnički interfejs.

4. ZAKLJUČAK

Zadatak ovog rada bio je da prikaže jezik za definisanje opisa podataka, sistem za konverziju veličina iz osnovnih u izvedene jedinice, podršku za višejezički korisnički interfejs i generator koji generiše osnovu projekta u skladu sa ovim definicijama. Iako je nastao kao rešenje za usko specificiran domen naftne industrije, dovoljno je generalizovan da bude upotrebljiv za bilo koju drugu aplikaciju slične arhitekture.

Omogućava definisanje opisa podataka, bez neophodnog programerskog znanja, obezbeđuje skladište tih podataka i pruža izgenerisanu osnovu projekta aplikacije koja koristi takvu definiciju podataka. Generator je zamišljen kao nezavisan međusloj između baze podataka i aplikacija koje je koriste. S obzirom da je projekat za koji je pre-vashodno implementiran velikog obima i da će se sastojati iz više samostalnih sistema, u budućnosti će biti proširen tako da obuhvata generisanje osnova aplikacija u drugim programskim jezicima i različitim platformama.

Prvobitna ideja i jeste da generator bude sprega između inženjera koji definišu model i programera koji treba na što brži i efikasniji način da dobiju model i osnovu projekta u obliku prilagođenom svojim potrebama. Budući da aplikacija sadrži pozamašan broj tabela, pored kojih se kontinuirano kreiraju nove, generator znatno ubrzava proces kreiranja odgovarajućih klasa entiteta i sloja za komunikaciju sa bazom podataka. Takođe otklanja mogućnost pravljenja ručno pisanih grešaka prilikom definisanja svojstava klasa i primenjivanje atributa koji određuju grupu jedinica kojima pripadaju svojstva, mapiranjem podataka iz XML dokumenta.

Dalja unapređivanja ovog alata mogla bi otpočeti proširivanjem generatora za podršku kreiranja osnove projekta na drugim programskim jezicima. Generator je trenutno samostalna aplikacija koja se mora pokrenuti kako bi se stekao željeni izlaz. U budućnosti bi se mogao napraviti okidač (*trigger*) kada dođe do izmene u bazi podataka koji će automatski pokrenuti generator i proizvesti novonastale ili modifikovane klase.

Što se tiče jezika specifičnog za domen koji trenutno rezultat opisa definicije podataka smešta u XML datoteku, trebalo bi ga proširiti opcijom za automatsko kreiranje i migriranje baze podataka.

5. LITERATURA

[1] U. Tikhonova, Engineering the Dynamic Semantics of Domain Specific Languages, 2017.

[2] I. Dejanović, Metamodel, editor modela i generator poslovnih aplikacija - Magistarska teza, Novi Sad, 2008.

[3] J. M. a. J. Mukerji, MDA Guide Version 1.0.1, 2003.

[4] I. Dejanović, Prezentacije sa predmeta Jezici specifični za domen.

[5] F. Tomassetti, A Guide to Code Generation, 2008.

Kratka biografija:

Ana Marojević rođena je 07.10.1994. u Novom Sadu. Fakultet tehničkih nauka upisuje 2013. godine, odsek Softversko inženjerstvo i informacione tehnologije. Diplomski rad odbranila je 2017. godine. Iste godine upisuje master akademske studije na Fakultetu tehničkih nauka.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.41
DOI: https://doi.org/10.24867/07BE39Blagojevic

AKTIVNI INKREMENTALNI GENERATOR GRAPHQL SERVERSKE APLIKACIJE

ACTIVE INCREMENTAL CODE GENERATOR FOR GRAPHQL SERVER APPLICATION

Bojan Blagojević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je predstavljen generator koda za server koji podržava GraphQL specifikaciju. Model se zadaje preko GraphQL SDL sheme. Generisanje koda se vrši uz pomoć obrađivača šablona. Podržano je i redefinisanje ponašanja aplikacije integracijom sa ručno pisanim kodom kao i inkrementalne izmjene sheme baze podataka.*

Ključne riječi: *Generator koda, GraphQL, SDL, šablon*

Abstract – *This paper presents the code generator for the web server that satisfies GraphQL API specification. The model is provided using GraphQL SDL schema. Code generation is performed using template engines. Generator also supports modifying application default behaviour by integrating generated code with manually written one. It also supports incremental updates for db schema.*

Keywords: *Code generator, GraphQL, SDL, template*

1. UVOD

Razvojem tehnologije te povećanjem broja aplikacija zasnovanih na mrežnoj komunikaciji, softverski inženjeri se suočavaju sa potrebom da definišu i unificiraju klijent-server komunikaciju. Kao proizvod ove ideje nastaje nekolicina specifikacija komunikacije između dvije udaljene tačke u vidu API-a (*Application programming interface*).

Među specifikacijama API-a, u posljednje vrijeme, najveću popularnost stekla je GraphQL specifikacija [1].

U prilog tome govori i činjenica preuzeta iz istraživanja [2] u kome se prema podacima iz 2017. godine svaki mjesec kreira preko 350 projekata na Github-u, u čijem se imenu ili opisu pominje riječ GraphQL.

Pisanje GraphQL servera se često svodi na pisanje repetitivnog koda vođenog unaprijed definisanom GraphQL shemom. Resursi opisani shemom su uglavnom dovoljni za definisanje interfejsa komunikacije. Nad navedenim resursima najčešće su definisane operacije dobavljanja, kreiranja, kao i njihovog brisanja i izmjene. Sve činjenice ukazuju na to da se softverski inženjeri dovode u situaciju da pišu ponavljajući kod, ali kako je on pisan od strane čovjeka, i dalje ostaje mogućnost pojave grešaka, nekonzistentnosti i bespotrebnog trošenja ljudskih resursa.

U idealnom slučaju GraphQL shema je dobro definisana na početku procesa razrade modela projekta.

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

Međutim, kako ovaj slučaj nije čest, shema je nerijetko podložna promjenama. Da je to zaista slučaj, potvrđuje i statistika izvučena iz analize evolucije sheme baze podataka za HMS softver koji je korišten u bolnicama u Velikoj Britaniji. Rezultati prikazani u radu [3], analizirani su tokom perioda od 1990. do 1991. godine.

Analizom se došlo do zaključka da je u navedenom periodu, broj relacija između entiteta porastao sa 23 na 55, što je uvećanje za 139%. Takođe, broj kolona se povećao sa 178 na 666, što predstavlja uvećanje za čak 274%. Interesantno je da je od 20 relacija koje su pronađene u novembru 1990. i 1991. godine, samo četiri su ostale nepromijenjene. U drugom iztraživanju [4] vršenom nad MediaWiki softverom koji je *backend* podrška za Wikipedia-u, autori rada su takođe napravili analizu izmena baze podataka. U studiji koja obuhvata četiri godine, došli su do zaključka da se broj tabela uvećao za 100%, a broj kolona za 142 %. Shema je pretrpjela i povećanje broja obrisanih atributa za 41.5 %, dok se broj preimenovanih polja povećao za 25.1 %. Sve ove promjene modela dodatno utiču na usporavanje procesa implementacije.

Jedno od rješenja ovog problema jeste uvođenje generatora koda za GraphQL server. Generatori koda se pišu sa ciljem da se isti generator koda može koristiti više puta za različite ulaze istog tipa. Pisanje generatora koda traje konstantno vrijeme, dok pisanje sličnih aplikacija koje generator može da izgeneriše ima linearnu kompleksnost koja zavisi od toga koliko se puta takva aplikacija implementira. Kada se jednom napiše generator koda za određeni programski jezik, veoma je lako dodati i *plugin* za generisanje iste pozadinske logike za drugi programski jezik. Generator koda rješava i problem konzistentnosti jer za iste programske konstrukcije koristi isti šablon te je na taj način jednoznačnost prilikom generisanja osigurana.

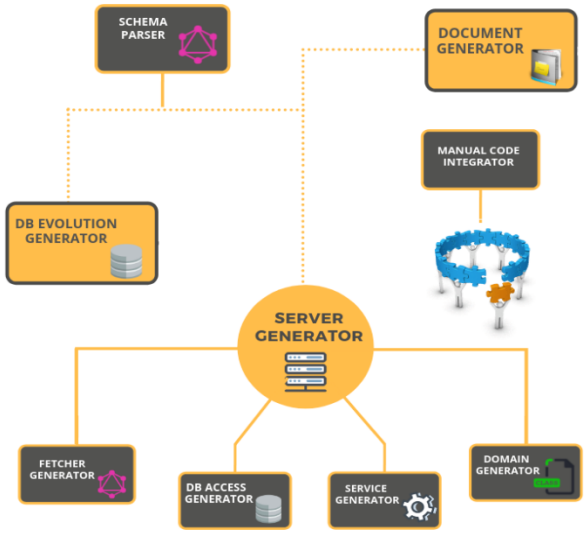
2. SPECIFIKACIJA SISTEMA

Problem kojim se ovaj rad bavi jeste mogućnost generisanja GraphQL servera na osnovu GraphQL sheme. Takođe, generator treba da riješi i problem čestih izmjena nad definisanom shemom, kao i da omogući redefinisanje standardnog ponašanja aplikacije tako što će da dozvoli korisniku da integriše ručno pisani kod. U nastavku teksta predstavljena je arhitektura generatora kroz prikaz njegovih osnovnih komponenti.

Generator GraphQL servera, koji se opisuje u radu, baziran je na pet osnovnih komponenti (Slika 1):

- Parser GraphQL sheme
- Generator evolucija baze podataka
- Generator servera
- Generator dokumentacije

• Komponenta za integrisanje ručno pisanog koda.



Slika 1. *Shema modela generatora*

Parser GraphQL sheme podataka predstavlja prvu komponentu u procesu generisanja. Ova komponenta je zadužena za parsiranje GraphQL sheme prema standardima koje definiše SDL format. Ulaz u parser sheme jeste jedna konkretna instanca SDL sheme preko koje su opisani svi entiteti koji će se koristiti u aplikaciji. Ova komponenta je zadužena i za detekciju grešaka nastalih u definiciji sheme čime se suzbija propagacija greške kroz druge komponente. Kao rezultat dobija se interna reprezentacija modela u vidu klasa kojima generator jednostavnije manipuliše. Interna reprezentacija (Slika 2) zapravo predstavlja međumodel koji služi kao ulaz u ostale komponente sistema.

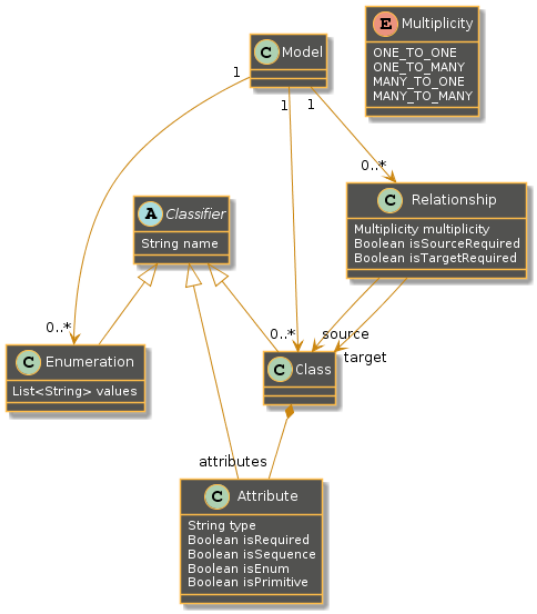
Generator evolucije baze podataka predstavlja komponentu zaduženu za generisanje sheme baze podataka na osnovu interne reprezentcije modela. Osnovne probleme koje treba da riješi ova komponenta jeste mapiranje GraphQL entiteta na entitete neke od relacionih baza podataka (Tabela 1), kao i prilagođavanje sheme baze podataka izmjenama nastalim u GraphQL shemi (Tabela 2).

Tabela 1. *Klase ekvivalencije između GraphQL tipova i relacione baze podataka*

GraphQL tipovi	**Relaciona baza podataka**
Type	*Table*
Primitive field	*Column*
Complex field	*Foreign key / Table*
Enum	*Enum type*

Tabela 2. *Mapiranje izmjena GraphQL sheme na evoluciju sheme baze podataka*

Izmjena GraphQL sheme	**Izmjena relacione sheme baze podataka**
Dodavanje novog tipa	*Dodavanje nove tabele*
Dodavanje novog primitivnog polja	*Izmjena tabele uz dodavanje nove kolone*
Preimenovanje polja	*Preimenovanje kolone tabele*
Promjena kardinaliteta sa oneToMany na manyToMany	*Dodavanje nove tabele uz brisanje kolone koja je predstavljala strani ključ u prethodnoj relaciji*



Slika 2. *Dijagram klasa interne reprezentacije modela*

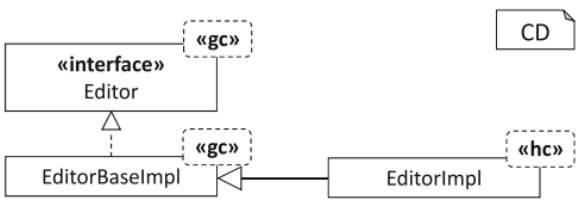
Najkompleksnija komponenta generatora je generator servera. Njegova osnovna funkcionalnost zasniva se na kreiranju funkcionalnog *backend*-a prema GraphQL specifikaciji. Sastoji se iz četiri međusobno zavisne cjeline:

- Generator domenskih klasa
- Generator servisnog sloja
- Generator GraphQL rukovaoca (eng. *handler*)
- Generator sloja za pristup bazi (*database access layer*)

Arhitekturalno gledano, prva tri generatora se mogu svrstati u sloj biznis logike aplikacije, dok se sloj za pristup bazi podataka može okarakterisati kao dio sloja za perzistenciju. Sloj koji je najbliži klijentskoj aplikaciji generiše se pomoću generatora za GraphQL *handler*-e. Za svaki od GraphQL tipova, generator GraphQL *handler*-a treba da omogući podršku za razrješavanje sljedećih zahtjeva:

- GraphQL mutacija za dodavanje entiteta
- GraphQL *query*-a za čitanje podataka

Postoji više tehnika za integraciju generisanog koda sa ručno pisanim kodom. U radu [5] definisane su tehnike za integraciju manuelno pisanog koda za generatore zasnovane na principima objektno-orijentisanog modela. Autori rada opisali su osam osnovnih tehnika za integraciju ručno pisanog koda evaluiranog kroz pet kriterijuma. Mehanizam koji je odabran da se koristi pri integrisanju ručno pisanog koda, autori rada su nazvali *Generation Gap*. Ova tehnika podrazumijeva da se za svaku operaciju generiše interfejs i uobičajeno ponašanje. Ukoliko korisnik želi da redefiniše ponašanje aplikacije, mora da kreira svoju implementaciju interfejsa koju će generator koristiti umjesto uobičajene implementacije. Model ovog mehanizma za integrisanje ručno pisanog koda prikazan je na slici 3.



Slika 3. *Generation Gap* [5] *mehanizam za integraciju ručno pisanog koda*

Generator dokumentacije, predstavlja posljednju komponentu sistema. Zadužen je za generisanje GraphQL IDE-a preko koga se može istraživati GraphQL shema i pregledati dokumentacija vezana za tipove i njihova polja. Takođe, IDE ima direktno integrisani *syntax highlighting* kao i validaciju operacija nad definisanom shemom. Pored toga, u navedenom IDE-u moguće je i izvršiti određene operacije nakon čijeg izvršavanja se prikazuje i rezultat. GraphQL IDE koji se generiše je web aplikacija integrisana sa izgenerisanim GraphQL serverom.

3. IMPLEMENTACIJA RJEŠENJA

3.1 Tehnologije

Generator je implementiran u Python programskom jeziku. Generisanje konstrukcija izvornog koda vrši se uz pomoć Python Jinja *template* jezika [6]. Izgenerisani kod pisan je u Scala programskom jeziku uz pomoć Play [7] radnog okvira. Za parsiraranje GraphQL sheme korištena je Python GraphQL biblioteka, dok se za serversku podršku GraphQL specifikacije u izgenerisanoj aplikaciji koristi Scala biblioteka pod imenom Sangria [8]. Perzistentni sloj prepušten je PostgreSQL bazi podataka, dok se za pristup istoj koristi Scala biblioteka pod imenom Slick [9]. Za pronalaženje razlika između trenutne i nove PostgreSQL sheme koristi se pgdiff [10] alat. Kompletna infrastruktura za lokalno okruženje podignuta je uz pomoć Docker [11] kontejnera.

3.2 Generator koda

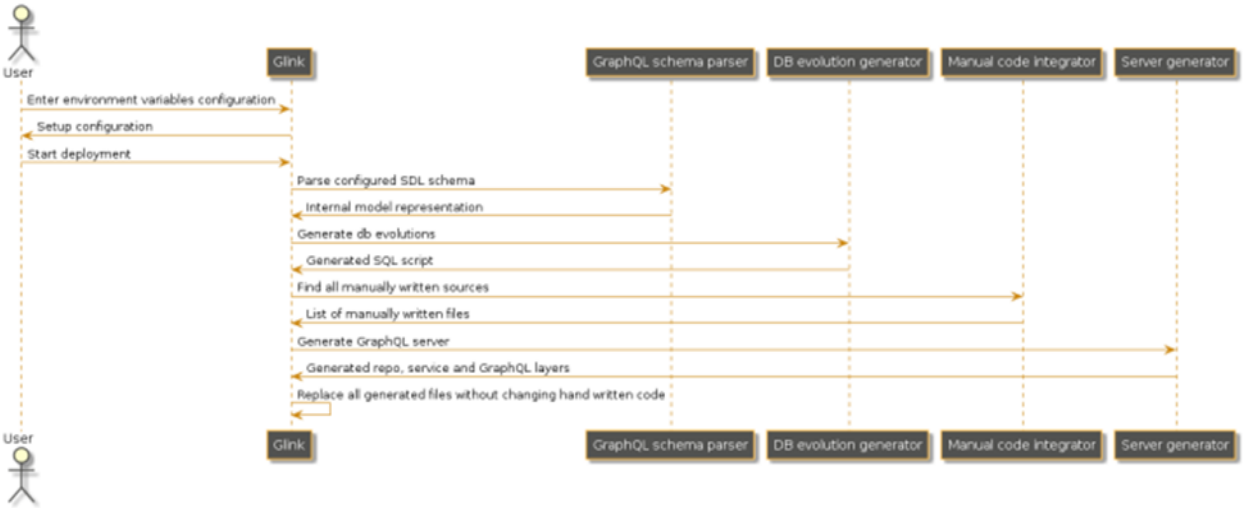
Proces generisanja GraphQL servera, kako je i prikazano na slici 4, sastoji se iz nekoliko koraka. Prvi dio u procesu generisanja predstavlja konfigurisanje putanje do izvorne sheme, foldera u kome će se nalaziti generisani kod i parametara za konekciju na PostgreSQL bazu podataka. Drugi dio u procesu generisanja ne zahtijeva dodatnu interakciju sa korisnikom generatora. Ovaj proces uključuje generisanje servera i svih propratnih sadržaja na osnovu unaprijed definisane konfiguracije. Kako bi se učinilo da čitav ovaj proces bude jednostavan za upotrebu, kreiran je i SDK preko koga se upotreba gorenavedenih instrukcija može olakšati pozivanjem jednostavnih komandi iz sistemske konzole. U skladu sa tim, SDK ima sljedeće komande:

- *init* – inicijalizacija projekta
- configure – podešavanje konfiguracije
- deploy – pokretanje procesa generisanja

Configure komanda služi za podešavanje parametara konfiguracije. Ova komanda može da prima dodatne parametre, za svaku varijablu po jedan. Mogu se konfigurisati putanja do SDL sheme, putanja do foldera u kome će se naći izgenerisani kod, kao i parametri za konekciju ka PostgreSQL bazi podataka.

Nakon što je konfiguracija projekta završena, korisnik generatora mora da pozove *deploy* komandu. Prvi proces koji se nakon toga dešava jeste proces pronalaženja i validiranja specificiranog SDL izvornog fajla. Validacija proslijeđene SDL sheme kao i njeno parsiranje vrši se korištenjem graphql-core Python biblioteke. API ove biblioteke pruža *parse* metodu, kojoj se prosljeđuje SDL shema, te se kao rezultat dobija *Document* objekat. Pimjer jedne ulazne GraphQL sheme dat je na listingu 1.

Generisanje sheme baze podataka u potpunosti je prepušteno Slick biblioteci. Slick biblioteka ima podršku za generisanje SQL iskaza za kreiranje sheme na osnovu opisa iste nad Scala klasama. Na taj način problem generisanja sheme, kao i njenog mapiranja, lokalizovan je na jednom mjestu. Ukoliko dođe do potrebe da generator podržava i neku drugu bazu podataka, neophodno je da se samo zamijeni *driver* baze podataka. Ostatak generisanja sheme je prepušten Slick implementaciji za navedeni *driver*.



Slika 4. *Dijagram sekvence za proces generisanja GraphQL servera*

```
type User {
  firstName: String!
  lastName: String
  email: String!
  age: Int
}
enum TodoState {
  CREATED
  IN_PROGRESS
  FINISHED
}

type Todo {
  title: String!
  description: String
  state: TodoState
  list: TodoList @oneToMany(mappedBy: "items")
}
type TodoList {
  title: String!
  items: [Todo] @manyToOne(mappedBy: "list")
  creator: User @oneToMany
}
```

Listing 1. *Primjer ulazne sheme generatora*

Nakon što se izgeneriše shema baze podataka, generiše se Play server. Generator serverske aplikacije zadužen je za generisanje Play aplikacije koja koristi GraphQL kao specifikaciju za razmjenu resursa. Generator generiše domenske klase aplikacije, sloj repozitorijuma, servisni sloj zajedno sa biznis logikom kao i GraphQL komponente (*GraphQL* tipovi, operacije i *resolver*-i). Kao posljednji proces u generisanju aplikacije izvršava se integracija ručno pisanog koda kao i generisanje GraphQL IDE-a.

4. ZAKLJUČAK

U ovom radu predstavljen je jedan jednostavan generator GraphQL servera namijenjen generisanju manjih aplikacija čiji domen ima uže područje djelovanja. Kao što je ranije navedeno, generišu se isključivo operacije za dobavljanje i kreiranje novih resursa, što za veće aplikacije često nije dovoljno. Velika primjena ovog generatora može se tražiti i u aplikacijama čiji je razvoj baziran na GraphQL shemi. Pod ovim se podrazumijevaju aplikacije u čijem se procesu razvoja prvo specificira struktura sheme, pa se onda implementacija prilagođava istoj. Takođe, generator može naći primjenu i u aplikacijama čiji je domen sklon čestim promjenama. Pošto generator automatizuje primjenu promjena sheme na bazu podataka kao i na izgenerisani kod, razvojni tim se može osloniti da rukovođenje promjenama prepusti generatoru. Na taj način smanjuje se potencijalni izvor grešaka kod ručno pisanog koda. Pošto generator generiše optimalan Scala kod u kombinaciji sa GraphQL konstrukcijama, namijenjen je i svim entuzijastima koji žele da nauče GraphQL tehnologiju.

Kako je navedeno u radu, generator može da generiše isključivo operacije za dobavljanje i kreiranje resursa. U određenim slučajevima ovakva postavka nije dovoljna te bi prvobitno poboljšanje moglo predstavljati dodavanje operacija za izmjenu i brisanje resursa. Pored toga, generator se može proširiti i dodavanjem dodatnih validacija polja kao što su jedinstvenost polja ili zadovoljavanje nekog šablon. Dodatni pravci proširenja mogu se ogledati i u dodavanju mehanizama za autorizaciju za određene operacije kao i generisanju korisničkog interfejsa aplikacije.

5. LITERATURA

[1] „GraphQL specification“, https://graphql.github.io/graphql-spec/June2018, pristupljeno: 11. jul 2019.

[2] „The GraphQL Data Language“, https://thenewstack.io/graphql-data-query-language-resource-guide, pristupljeno: 11. jul 2019.

[3] Dag Sjoberg, Lilybank Gardens, „Quantifying Schema Evolution“, Information and Software Technology, Vol. 35, No. 1, pp. 35-44, januar 1993.

[4] Carlo A. Curino, Hyun J. Moon, Letizia Tanca, Carlo Zaniolo, „Schema evolution in Wikipedia toward a Web Information System Benchmark“, ICEIS 2008 - Proceedings of the Tenth International Conference on Enterprise Information Systems, Volume DISI, 12-16 jun 2008.

[5] P. Desfray, J. Filipe, S. Hammoudi, L. Pires, „Integration of Handwritten and Generated Object-Oriented Code“, Model-Driven Engineering and Software Development, Communications in Computer and Information Science, vol 580. Springer, Cham, 2015.

[6] „Jinja2“, https://jinja.palletsprojects.com/en/2.10.x/, pristupljeno: 17. avgust 2019.

[7] „Play framework“, https://www.playframework.com, pristupljeno: 17. avgust 2019.

[8] „Sangria“, https://sangria-graphql.org, pristupljeno: 17. avgust 2019.

[9] „Slick“, http://slick.lightbend.com/, pristupljeno: 17. avgust 2019.

[10] „pgdiff“, http://pgdiff.sourceforge.net/, pristupljeno: 17. avgust 2019.

[11] „Docker“, https://www.docker.com, pristupljeno: 17. avgust 2019.

Kratka biografija:

Bojan Blagojević rođen je 1. januara 1994. godine u Bijeljini, Republika Srpska, Bosna i Hercegovina. Osnovnu školu „Aleksa Šantić” u Ugljeviku i Gimnaziju „Filip Višnjić” u Bijeljini je završio kao nosilac Vukove diplome. 2013. godine upisao je Fakultet tehničkih nauka u Novom Sadu, smijer Softversko inženjerstvo i informacione tehnologije. 2017. godine završava pomenute osnovne akademske studije sa prosječnom ocjenom 10.00. Iste godine upisuje master akademske studije na istom smijeru. Položio je sve ispite predviđene planom i programom.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE40Ilijazovic

КЛАСТЕРИЗАЦИЈА ВИСОКОДИМЕНЗИОНАЛНИХ ПОДАТАКА ЕЛЕКТРОЕНЕРГЕТСКИХ ПОТРОШАЧА У ПРОГРАМСКОМ ЈЕЗИКУ R

HIGH-DIMENSIONAL CLUSTERING OF ELECTRIC CONSUMER DATA IN R PROGRAMMING LANGUAGE

Елведин Илијазовић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *Тема рада је поређење перформанси софтверског решења за кластеризацију профила потрошње у програмском језику R у односу на решење имплементирано у .NET развојном окружењу. У раду је описан програмски језик R и његове библиотеке, које су коришћене у реализацији решења. Описани су недостаци због којих се јавља потреба за кластеризацијом профила као и метода редуковања димензионалности података уз минимални губитак информација, а све зарад тачности саме кластеризације.*

Кључне Речи: *Кластеризација, машинско учење, програмски језик R*

Abstract – *In this work it will be shown how data clustering of Consumer Load Profiles in R programming language performs, opposed to same implementation in .NET framework. Programming language R alongside with its libraries that have been used for this implementation has been described. The reason of why data clustering is used has also been described with methods of machine learning.*

Keywords: *Clustering, Machine Learning, R Programming Language*

1. УВОД

Напретком технологије имамо могућност складиштења података о потрошњи електричне енергије на нивоу појединачних потрошача и у релативно честим временским интервалима. На основу ових података о понашању потрошача могуће је естимирати производњу електричне енергије и смањити губитке Прикупљени подаци су у сировој форми и, као такви, нису подобни за коришћење.

Један од разлога је сама количина података те их је потребно груписати по сличности како би били корисни у аналитичким процесима система управљања дистрибуцијом.

У овом раду је описана кластеризација података о електричној потрошњи у програмском језику R. Приказани су алгоритми кластеризације од којих је детаљно описан ***K-means***.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Александар Купусинац, ванр.проф.

Уводни део описа решаваног проблема описује потребу за прикупљањем података као и потребу за њиховом кластеризацијом. Описани су профили потрошње, као и потреба за прикупљањем не само мерења о активној снази потрошача, већ и мерења о његовој реактивној снази.

Такође, описана је анализа главних компоненти у циљу редуковања димензионалности простора са подацима као и алгоритам најближих суседа у циљу класификације потрошача са невалидним сетом података.

Поглавље са описом постојећег решења садржи детаљан опис формирања сета података који се кластерују као и процес самог кластеровања.

У раду су описане коришћене технологије, пре свега програмски језик ***R*** који је служио за имплементацију решења овог рада као и ***.NET*** развојно окружење у којем је имплементирано постојеће решење. Такође је описана ***Microsoft SQL Server*** релациона база података која служи за складиштење података.

Приликом описа решења приказани су исечци R кода са методама које су позиване, њихов опис као и опис параметара које им се прослеђују. Упоређено је постојеће решење имплементирано у .NET развојном окружењу са решењем овог рада. Измерене су перформансе између ове две апликације које су вршиле кластеризацију података са идентичном поставком улазних параметара.

2. ОПИС РЕШАВАНОГ ПРОБЛЕМА

У овом поглављу биће описани проблеми са којом се суочава електродистрибуциона мрежа. Такође ће бити описани профили потрошње као и потреба за њиховом кластеризацијом. Дискутовано је постојеће решење као и алгоритми кластеризације, њихова валидација, редукција димензионалности и алгоритам класификације.

2.1 СНАБДЕВАЊЕ И ПОТРАЖЊА ЕНЕРГИЈЕ

У електроенергетском систему баланс између генерисања електричне енергије и њене потрошње (у потрошњу се урачунава и енергија губитка на мрежи) представља важан аспект. Веома је важно да снабдевена генерисана енергија одговара потрошњи, јер у супротном долази до нестабилности електричне мреже, што може довести до прекида снабдевања енергијом. Како енергетски систем спада у систем са

критичном мисијом, ово може имати значајне последице јер утиче на животе великог броја људи.

Електроенергетске компаније овај проблем решавају проценом потражње електричне енергије и на основу ове естимације планирају производњу. Потрошња се естимира на основу претходне потрошње, а за то су нам потребна мерења потрошача на основу којих се креирају профили потрошње.

2.2 ПРОФИЛИ ПОТРОШЊЕ

У електроинжењерству, профил потрошње представља граф промене електричне потражње односно потрошње по специфичном времену. Произвођачи електричне енергије користе ове информације како би испланирали колико електричне енергије ће бити потребно обезбедити за неко време у будућности.

Прикупљањем података о потрошњи сваког потрошача јавља се нови проблем – обим података. Уколико се користи петнаестоминутни интервал, за измерене вредности активне и реактивне снаге за период од једне године, може настати и до неколико десетина, па чак и стотина гигабајта података. Ова огромна количина података није погодна за рачунарску обраду у реалном времену, па се пре тога ови подаци агрегирају у профиле потрошње и над њима се врши кластеризација како би се редуковао скуп података са што мањим губитком информација.

2.5 КЛАСТЕРИЗАЦИЈА

Кластеризација представља груписање објеката на такав начин да су објекти у групи (кластеру) више слични (у неком смислу) међусобно, у односу на објекте група којима не припадају [1]. Ово је главни задатак *data mining-a* у статистичкој анализи података. За решење нашег проблема одабран је модел центроида. Потрошачки профили који припадају истој групи добијају вредност средње вредности свих чланова групе и тиме се редукује количина података потребна за прорачуне.

2.5.1 *K-MEANS* АЛГОРИТАМ

Метода средњих вредности (***k-means***) је метода векторске квантизације која је популарна за кластеризацију приликом анализе података. Ова метода има за циљ да распореди $\boldsymbol{N}$ тачака $\boldsymbol{I}$-димензионалног простора у $\boldsymbol{k}$ група, односно кластера, у којој свака тачка припада кластеру са најсличнијим карактеристикама. Тачке означавамо са $\{\boldsymbol{x}^{(n)}\}$, где се $\boldsymbol{n}$ креће од 1 до броја тачака $\boldsymbol{N}$. Претпоставка је да је простор у коме се налази $\boldsymbol{x}$ реалан и да постоји метрика којом се дефинише дистанца између тачака, као на пример једначина за израчунавање дистанце у еуклидском простору:

$$d(x,y) = \sqrt{\sum_i (x_i - y_i)^2} \qquad (1)$$

Најчешћа коришћена верзија алгоритма користи итеративну технику за проналажење кластера. Овај алгоритам се може поделити у три корака: корак иницијализације, корак доделе и корак ажурирања.

У кораку иницијализације се постављају иницијалне средње вредности $\{\mathbf{m}^{(k)}\}$ за кластере. Иницијалне средње вредности кластера се могу изабрати насумично, али постоје и посебне методе за иницијализацију.

У кораку доделе, свака тачка $\boldsymbol{n}$ се додељује кластеру чијем је центру најближа по Еуклидском растојању. Означимо нашу претпоставку да тачка $\boldsymbol{x}^{(n)}$ припада кластеру $\boldsymbol{k}^{(n)}$ као $\widehat{\boldsymbol{k}}^{(n)}$ тако да:

$$\hat{k}^{(n)} = \underset{k}{\operatorname{argmin}}\left\{d(\mathrm{m}^{(k)}, x^{(n)})\right\} \qquad (2)$$

Алтернативно, можемо ову доделу тачака кластерима представити и кроз параметре „задужења“ $\boldsymbol{r}_{\boldsymbol{k}}^{(\boldsymbol{n})}$. У кораку доделе параметеру $\boldsymbol{r}_{\boldsymbol{k}}^{(\boldsymbol{n})}$ се додељује вредност 1 ако је средња вредност кластера $\boldsymbol{k}$ најближа тачки $\boldsymbol{x}^{(\boldsymbol{n})}$, у супротном $\boldsymbol{r}_{\boldsymbol{k}}^{(\boldsymbol{n})}$ добија вредност 0. Ово је представљено једначином:

$$r_k^{(n)} = \begin{cases} 1, & \hat{k}^{(n)} = k \\ 0, & \hat{k}^{(n)} \neq k \end{cases} \qquad (3)$$

Након корака доделе следи корак ажурирања, у коме се рачунају нове позиције центара кластера које представљају средње вредности тачака које су му додељене тј. за које је задужен.

$$\mathrm{m}^k = \frac{\sum_n r_k^{(n)} x^{(n)}}{R^{(k)}} \qquad (4)$$

где је $R^{(k)}$ укупно задужење средње вредности k, и једначина му је:

$$R^{(k)} = \sum_n r_k^{(n)} \qquad (5)$$

Кораци доделе и ажурирања се понављају све до тренутка када задужења за све вредности кластера не мењају, тј. када алгоритам конвергира.

Алгоритам методе средњих вредности не гарантује да ће увек конвергирати глобалном оптимуму. Резултат кластеровања увелико зависи од почетне иницијализације средњих вредности кластера. Како је овај алгоритам у већини случајева брз, честа је пракса да се покрене више пута са различитим сетом иницијалних вредности.

2.5.2 *K-MEANS++* АЛГОРИТАМ

У анализи података, ***k-means++*** представља алгоритам за одабир иницијалних вредности који се користе као улаз у методу средњих вредности. Овај алгоритам је предложен 2007. године од стране Дејвида Артура и Сергеја Василвитског, у виду алгоритма апроксимације за НП тешке проблеме кластеровања по принципу средњих вредности као начин за избегавање повремено лошег кластеровања од стране стандардног алгоритма средњих вредности.

Овај алгоритам креће са интуицијом да је боље раширити позиције иницијалних центара кластера по простору са подацима. Први центар је насумично одабран из објеката који се групишу, после чега се сваки наредни центар бира од остатака објеката са вероватноћом пропорционалном његовој квадратној дистанци од најближег постојећег центра кластера.

2.5.3 ИНДЕКСИ ВАЛИДНОСТИ

Индекс валидности кластеровања служи за мерење квалитета резултата кластеровања у поређењу са резултатима добијеним другим алгоритмом кластеризације, или истим алгоритмом са другачијим улазним параметрима. Ови индекси су обично погодни за мерење кластера које се међусобно не преклапају [2].

2.5.4 АНАЛИЗА ГЛАВНИХ КОМПОНЕНТИ *(PCA)*

Кластеризација производи лошије резултате што је број димензија већи. Из тог разлога је потребно редуковати димензионалност на такав начин да се и даље очува већина информација о подацима. Ово се постиже помоћу анализе главних компоненти.

Анализа главних компоненти (***Principal component analysis*** (***PCA***)) представља процедуру у статистици која користи ортогоналну трансформацију у циљу конверзије сета објеката(обсервација), са могућим узајамним везама између варијабли (ентитети који имају неку нумерчку репрезентацију), у сет вредности линеарно неповезаних варијабли које се зову главне компоненте (***principal conponents***).

PCA може бити замишљен као процес постављања ***n*** димензионог елипосоида подацима, где свака оса елипсоида представља главну компоненту. Уколико су неке осе елипсоида мале, тада је варијанса по тој оси такође мала и уклањањем те осе и главних компоненти које им припадају долази до малог губитка информација.

2.5.5 АЛГОРИТАМ К НАЈБЛИЖИХ СУСЕДА

У препознавању узорака (***pattern recognition***), алгоритам к најближих суседа (***k-nearest neighbours*** (***k-NN***)) се користи као непараметризована метода за класификацију и регресију [3]. У оба случаја, улаз се састоји од *k* најближих тест примера у функцији простора.

2.4 ПОСТОЈЕЋЕ РЕШЕЊЕ

Постојеће решење је имплементирано у ***.NET*** развојном окружењу и изведено је у два корака, корак претпроцесирања, који служи за припрему података и корак процесирања, који врши кластеризацију над подацима.

2.4.1 КОРАК ПРЕТПРОЦЕСИРАЊА

Овај корак неће бити имплементиран у *R* језику, али је описан како би се приказала структура података који се кластерују.

У овом кораку се креирају потрошачки профили за сваког потрошача на основу достављених података и конфигурације. Конфигурација се састоји из годишњих сезона и типичних дана(нпр. радни дан, викенд). Подаци се агрегирају тако што се израчуна просек потрошње који је измерен код потрошача за дати временски тренутак за сваку комбинацију конфигурисаних сезона и типичних дана. Свако мерење представља једну димензију простора по којој се подаци кластерују. Број димензија зависи од итнервала мерења и конфигурације, и добија се множењем броја мерења у једном дану, бројем сезона, бројем типичних дана и врстом мерења (мере се и активна и реактивна снага за дати трентуак). Тако се за нпр. 48 дневних мерења, 12 сезона и 3 типична дана добије 3.456 димензија.

Такође се у овом кораку врши нормализација на начин да се све вредности активне снаге потрошача поделе са његовом просечном годишњом активном снагом. Излаз овог корака су нормализоване агрегиране криве намапиране на одговарајућу комбинацију конфигурационих параметара.

2.4.2 КОРАК ПРОЦЕСИРАЊА

У овом кораку се врши кластеризација над подацима генерисаним у претпроцесирању. Пре самог кластеровања потребно је, за сваког потрошача, све комбинације кривих спојити у једну. Врши се и селекција потрошача на валидне и невалидне. Валидни потрошачи су они који садрже криве за сваку комбинацију, а остали се прогласе за невалидне и они ће бити класификовани у цетроиде креиране на основу валидних потрошача.

Постојеће решење за алгоритам кластеровања користи *k-means++* који као улазни параметар и, поред самих података, захтева и број кластера. Како нам је податак о улазним параметрима непознат, задаје се опсег група од минималног до максималног броја кластера. Алгоритам се извршава за сваки број група из опсега и узима се најбоље решење. Одабир најбољег решења врши се уз помоћ индекса валидности кластера.

Излаз овог корака су центроиде кластера (потрошачке групе) као и њихово мапирање на потрошаче који им припадају. Ово значи да типичну криву неког потрошача добијамо множењем његовог потрошачког профила са средњим годишњим снагама.

3. ПРОГРАМСКИ ЈЕЗИК *R*

R је програмски језик и бесплатно софтверско окружење за статистичке прорачуне и графику које подржава ***R*** фондација. ***R*** језик је широко коришћен међу статистичарима за развој статистичког софтвера и анализу података. Овај програмски језик омогућава компајлирање и покретање на великом броју ***UNIX*** платформи, као и на ***Windows*** и ***MacOS*** оперативним системима. Анкете, анализе података и студије научне литературе показују значајно повећање популарности овог програмског језика. Од октобра 2019. године, ***R*** је на 15. месту по ***TIOBE*** индексу мерења популарности програмских језика [4].

4. *R* ИМПЛЕМЕНТАЦИЈА РЕШЕЊА

Први корак у апликацији је добављање нормализованих дневних профила потрошње из базе података које су креиране у кораку претпроцесирања. Калстеризација се обавља над валидним потрошачима. На основу ових података генерише се матрица података са редовима потрошача и колонама спојених комбинација нормализованих мерења. Ова матрица, на реалном примеру са 12 сезона, 3 карактеристична дана и петнаестоминутним интер-валом (96 дневних мерења), садржи 6.912 колона (димензија).

Следећи корак је редукција димензионалности помоћу ***PCA*** алгоритма. Ово се постиже помоћу ***prcomp***

функције која врши анализу главних компоненти улазне матрице података.

```
PCA.result = prcomp(data, center = TRUE, scale = TRUE)
```

Одабир главних компоненти које садрже 95% информација је имплементиран на следећи начин:

```
#Računanje varijanse:
variance = PCA.result$sdev ^ 2
#Računanje kumulativnih proporcija:
variance.proportion = cumsum(variance / sum(variance))
#Odabir dimenzija koje sadrže 95% informacija:
reduced.dim = GetRowCount
(variance.proportion[variance.proportion <= 0.95]) + 1
#Dimenzije koje su nam od značaja:
PCA.result$x[, 1:reduced.dim]
```

Након редукције димензија следи корак кластеровања. Ово је постигнуто помоћу ***KMeans_rcpp*** функције из пакета ***ClusterR***:

```
ClusterR::KMeans_rcpp(data = PCA.result,
                      clusters = k,
                      num_init = n,
                      initializer = "kmeans++",
                      seed = sample(1:10000, 1))
```

Резултат функције ***KMeans_rcpp*** је мапирање објекта са групом којој припада.

Након завршетка кластеризације, врши се израчунавање индекса валидности. За индекс валидности се користи Дејвис-Боулдин индекс валидности из ***clusterSim*** пакета. Позив функције је следећи:

```
clusterSim::index.DB(data,
                     cl=clusters,
                     d=NULL,
                     centrotypes="centroids",
                     p=2,
                     q=2)$DB
```

Калстеризација се извршава за опсег група ***kmin*** - ***kmax*** са ***n*** понављања и као резултат се узима она вредност кластеризације која има најбољи индекс валидности.

Наредни корак је класификација потрошача са невалидним мерењима, тј. потрошача који немају све податке за комбинацију сезона и карактеристичних дана. Ово се постиже помоћу функције ***predict.***

Добијен резултат апликације је матрица са центроидама група и мапирање потрошача на групу којој припада.

5. ПОРЕЂЕЊЕ СА ПОСТОЈЕЋИМ РЕШЕЊЕМ

Два решења у овом раду су поређена над истим скупом података и са истим улазним конфигурационим параметрима и извршена су над истом машином. Тестирање се извршавало над скупом од 11.494 потрошача, од којих су 8.055 валидни, а 3.494 невалидни. Број димензија за задату комбинацију конфигурације износи 6.912. Опсег група је 50-100 и број понављања 3. Резултат итвршавања апликација је приказан у табели 1.

Табела 1. *Поређење времена извршавања*

	.NET имплементација	R имплементација
Време трајања PCA алгоритма	03:16:12	00:04:43
Број димензија након редукције	64	205
Време трајања кластеризације	07:48:33	2:15:05
Број кластера	53	54
Укупно средње време извршавања	11:04:45	02:19:48

Из табеле 4.5 се може закључити да је време проналажења кластера ***R*** имплементације за 79% краће у односу на време проналажења ***.NET*** имплементације.

6. ЗАКЉУЧАК

У раду је приказано решење за кластеризацију које је имплементирано у програмском језику ***R.*** Приказано решење је имало боље перформансе у односу на постојеће решење имплементирано у ***.NET*** развојном окружењу. Иако ***PCA*** алгоритам у ***R*** имплементацији редукује димензионалност на 205 димензија, у односу на постојеће решење које димензионалност смањује на 64 димензије, ***R*** имплементација и даље долази до резултата у краћем временском интервалу.

Треба напоменути и лакоћу имплементације. Намена ***R*** програмског језика је управо у статистичким прорачунима над подацима и као такав садржи већ имплементиране библиотеке које су се користиле у нашој имплементацији. Постојеће (***.NET***) решење је морало имплементирати све алгоритме наведене у овом раду. Ово значи да је имплементација у ***R*** језику неупоредиво једноставнија и, самим тим што садржи мање кода, једноставнија за одржавање и евентуалне измене и унапређења.

7. ЛИТЕРАТУРА

[1] S. Guha, R. Rastogi and K. Shim: *CURE: an efficient clustering algorithm for large databases*, Proc. of ACM SIGMOD International Conference on Management of Data, pp. 73 – 84, 1998.

[2] Csaba Legány, Sándor Juhász and Attila Babos; "*Cluster Validity Measurement Techniques*"; Proceedings of the 5th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Madrid, Spain, February 15-17, 2006 (pp388-393)

[3] Altman, N. S. (1992). "*An introduction to kernel and nearest-neighbor nonparametric regression*". The American Statistician. 46 (3): 175–185. doi:10.1080/00031305.1992.10475879. hdl:1813/31637

[4] TIOBE Index - The Software Quality Company, TIOBE, https://www.tiobe.com/tiobe-index/ (приступљено октобра 2019.)

Кратка биографија:

Елведин Илијазовић рођен је у Новом Саду 1992. год. Факултет техничких наука уписао је 2011. год. Бечелор рад из области Електротехнике и рачунарства – рачунарске науке и информатика, одбранио је 2016. год.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE41Salaji

IMPLEMENTACIJA MEHANIZMA ZA RAZMENU PODATAKA PO UGLEDU NA *Apache Kafka* ARHITEKTURU

IMPLEMENTATION OF DATA SHARING MECHANISM MODELED ON *Apache Kafka* ARCHITECTURE

Kristijan Salaji, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je predstavljeno programsko rešenje za sinhronu i asinhronu razmenu podataka u distribuiranim sistemima po ugledu na Apache Kafka arhitekturu. Pored opisa implementacije rešenja, u ovom radu je implementirana i replikacija podataka.*

Ključne reči: *Kafka, proizvođač, potrošač, posrednik*

Abstract – *This paper presents a software solution for synchronous and asynchronous data exchange in distributed systems modeled on Apache Kafka architecture. In adition to concept of implementation, there is an implementation of replication service.*

Keyords: *Kafka, producer, consumer, broker*

1. UVOD

1.1 Distribuirani sistemi

Distribuirani sistem jeste skup nezavisnih računara koje korisnici vide kao jedan koherentan sistem [1]. Svi ti računari, povezani računarskom mrežom, rade zajedno u cilju izvršavanja zajedničkog zadatka.

Razlozi za razvoj distribuiranih sistema su sledeći [1]:

- Jednostavnije pristupanje resursima – jedan od glavnih ciljeva distribuiranih sistema jeste da obezbedi korisnicima lak pristup udaljenim resursima
- Transparentnost – važno je sakriti činjenicu da se procesi i resursi nalaze na više distribuiranih mašina
- Otvorenost – distribuirani sistem izlaže svoje servise po unapred definisanim standardima i pravilima
- Skalabilnost – dodavanje novih korisnika i resursa u sistem.

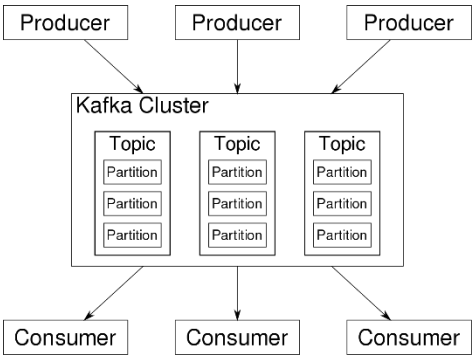
1.2 *Apache Kafka*

Apache Kafka je softverska platforma za obradu podataka koja ima za cilj da obezbedi prenos podataka u realnom vremenu, sa visokom tačnošću i minimalnim kašnjenjem [2]. *Apache Kafka* je zasnovana na dnevniku objava (*Commit Log*), koji omogućava korisnicima da se pretplate na neku od objava, ili da sami objave podatke dostupne bilo kom od računara u Kafka sistemu i aplikacijama koje rade u realnom vremenu [2].

Softversko rešenje izloženo u radu je bazirano na *Apache Kafka* arhitekturi (Slika 1.1). Razvijeni programski sistem za distribuciju poruka u celosti je saglasan Kafka modelu.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Branislav Atlagić, docent.

Pojednostavljena *Kafka* arhitektura sastoji se od klastera (*Cluster*), proizvođača (*Producer*) i potrošača (*Consumer*).



Slika 1.1. *Kafka arhitektura [3]*

Kafka klaster se sastoji od više posrednika (*Broker*). Proizvođač šalje podatke posredniku koji iste skladišti u bazu podataka. Potrošač šalje zahteve posredniku kada je spreman da prihvati i obradi potrebne podatke.

1.2.1 *Kafka* teme i particije

U okviru *Kafka* arhitekture, svaki podatak objavljen od strane bilo kog proizvođača vezan je za određenu temu (*Topic*). Podaci vezani za jednu temu se čuvaju na više particija koje mogu da se nalaze na različitim posrednicima [4]. Jedna od glavnih uloga particije jeste da se ubrza rukovanje podacima, uvođenjem paralelizma u radu. Podaci se istovremeno upisuju na više particija, kao što se mogu i čitati [3].

1.2.2 *Kafka* proizvođač

U *Kafka* arhitekturi proizvođač se poklapa sa izdavačem (*Publisher*) u *Publish/Subscribe* arhitekturi. Proizvođač predstavlja izvor podataka koji se upotrebljavaju u celokupnom *Kafka* sistemu. Podaci se objavljuju na postojeće teme i kasnije čuvaju u particijama u sklopu posrednika.

1.2.3 *Kafka* potrošač

Potrošači čitaju podatke sa željenih tema koji se nalaze u sklopu posrednika. Glavna razlika koja se uvodi u *Kafka* arhitekturi jeste ta da potrošači neće automatski dobijati podatke sa tema koji su im od interesa. Kada su spremni da obrade podatke, potrošači će sami poslati zahtev za podacima [3].

1.2.4 *Kafka* grupe potrošača

Grupa potrošača (*Consumer Groups*) se sastoji od jednog ili više potrošača. U većini slučajeva je jedna grupa potrošača vezana za jednu temu. Svakom potrošaču u grupi dodeljuje se jedna ili više particija određene teme [3]. Ova podela potrošača u grupe omogućava paralelizam u radu i ubrzava proces dobavljanja i obrade potrebnih podataka.

1.2.5 *Kafka* posrednik

Posrednik (*Broker*) se nalazi u okviru *Kafka* klastera. Posrednik može da vodi računa o više particija [3]. Primarni zadatak posrednika jeste da prima podatke, pretvara ih u zapise kojima dodeljuje *Offset* i nakon toga zapise smešta u particije. Particije vezane za jednu temu mogu biti raspoređene na nekoliko posrednika [3].

1.2.6 *Kafka* klaster

Kafka klaster (*Cluster*) se sastoji od više posrednika. U okviru svakog klastera jedan posrednik se proglašava za upravljača (*Controller*).

Njegova uloga jeste da dodeljuje particije ostalim posrednicima, obavlja administrativne operacije i vodi računa o otkazima preostalih posrednika [3].

2.TEORIJSKE OSNOVE

2.1 Replikacioni servis

Poravnavanje podataka na više instanci istog servisa se naziva replikacija podataka. Replikacijom podataka obezbeđuje se konzistentnost, povećava nivo sigurnosti sistema i smanjuje verovatnoća gubitka podataka usled pada sistema.

Replikacioni servis omogućava jednostavno kopiranje podataka na jednu ili više lokacija. Za replikacioni servis su vezani partnerski replikacioni servisi i komponenta koja replicira ili dobija replikacione podatke.

2.2 *Failover*

Failover predstavlja operativni režim rada gde sekundarna komponenta preuzima funkcije sistema kada primarna komponenta postane nedostupna zbog kvara ili unapred planiranih prekida rada [7].

Stanja u kojima se nalaze komponente sistema koji podržava failover:

- *Hot* – komponenta koja se nalazi u *Hot* stanju jeste primarna komponenta i ona izvršava funkcije sistema u datom trenutku
- *StandBy* – sekundarna komponenta jeste komponenta koja se nalazi u *StandBy* stanju. Ova komponenta je u svakom trenutku u pripravnosti da preuzme funkcije sistema ukoliko primarna komponenta nije u mogućnosti da izvršava iste

Sistem koji podržava *Failover* jeste sistem čija je otpornost na greške na visokom nivou. *Failover* jeste sastavni deo sistema sa kritičnom misijom koji moraju biti dostupni u svakom trenutku [7]. Postupak prebacivanja funkcijonalnosti na sekundarnu komponentu treba da bude što je više moguće neprimetan za krajnjeg korisnika [7].

3.OPIS KORIŠĆENIH TEHNOLOGIJA I ALATA

3.1 *.NET framework*

.NET framework služi za razvoj i izvršavanje aplikacija na *Windows* operativnim sistemima. *.NET framework* je deo *.NET* platforme, koja predstavlja kolekciju tehnologija za razvoj aplikacija na različitim operativnim sistemima. Komponente koje čine jezgro *.NET framewrok-a su* [5]:

- *Common Language Runtime* (CLR)
- *.NET Framewrok Class Library* (FCL)

3.2 C# programski jezik

C# (C *Sharp*) je objektno orjentisan programski jezik i čini sastavni deo *.NET framework-a*. Jedan je od mlađih programskih jezika i veoma je sličan Java programskom jeziku. C# kombinuje računarsku moć C++ sa lakoćom programiranja koju poseduje *Visual Basic* [6].

3.3 *Microsoft Visual Studio*

Za razvoj projekta opisanog u ovom dokumentu korišćen je *Microsoft Visual Studio*, integrisano razvojno okruženje (IDE). *Visual Studio* ima široku upotrebu prilikom razvoja *web* aplikacija i *web* servisa, *web* sajtova, i drugih računarskih programa.

U okviru *Visual Studio-a* se nalazi alat *Performance Profiler*. Pomoću njega se mogu meriti performanse sistema, zauzeće memorije, broj poziva funkcija, vreme provedeno u funkcijama i ostale korisne stvari koje mogu pomoći prilikom unapređenja performansi razvijene aplikacije.

3.4 *NUnit*

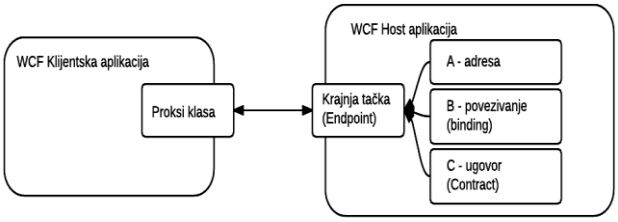
Testiranje ispravnosti napisanog koda vrši se pomoću automatskih testova od kojih neki testiraju pojedinačne klase (*unit* testovi) a drugi čitave komponente (*component* testovi). *Unit* testovi predstavljaju manje celine koda napisane od strane testera ili programera. Pomoću ovih testova proveravaju se delovi koda koji imaju jedinstvenu funkcionalnost, kao što su jedna metoda ili funkcija.

3.5 *Open Cover*

Za merenje procenta pokrivensti koda testovima koristi se alat pod nazivom *Open Cover*. *Open Cover* je besplatan alat koji može da se primenjuje na *.NET* 2.0 i novijim verzijama. Pomoću ovog alata možemo da pratimo pokrivenost grana kao i pokrivenost određenih sekvenci koda.

3.6 *Windows Communication Foundation* (WCF)

Windows Communication Foundation (WCF) jeste model za razmenu podataka koji omogućava komunikaciju preko mreže ili lokalnu komunikaciju. WCF uključuje set biblioteka koje su razvijene za distribuirano programiranje [8].



Slika 3.1. *WCF dijagram [8]*

3.7 Generički tipovi podataka

Generički tipovi podataka su uvedeni u C# verziji 2.0. Oni omogućavaju da se definiše klasa sa rezervisanim mestom za tip polja, tip povratne vrednosti funkcije ili tip parametra u samoj funkciji [9]. Generički tipovi podataka

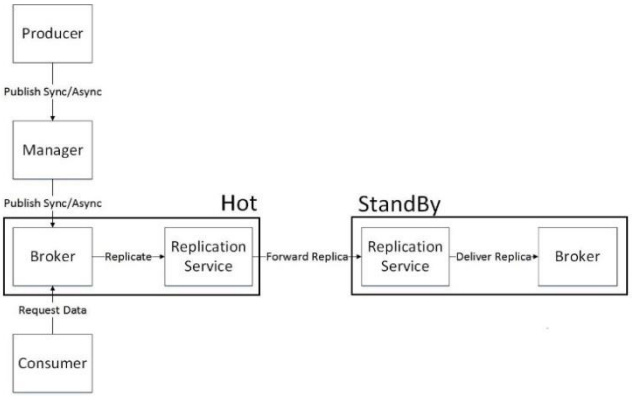
se najviše upotrebljavaju kod metoda koje rukuju sa kolekcijama [10].

3.8 *Callback Handler*

Callback je interfejs definisan kao atribut drugog interfejsa koji izlaže servis. *Callback* interfejs treba da implementira klijent koji ostvaruje vezu sa serverom. *Callback* se koristi kada server treba da obavesti klijenta da se desio neki događaj ili da isporuči klijentu neke podatke bez potrebe da klijent kreira *Host.*

4. IMPLEMENTACIJA PROGRAMSKOG REŠENJA

4.1 Arhitektura realizovanog rešenja



Slika 4.1 - *Arhitektura implementiranog rešenja*

Slika 4.1 prikazuje arhitekturu implementiranog rešenja. Prethodno navedene komponente čine mehanizam za razmenu podataka po ugledu na *Kafka* rešenje. Na slici je prikazana *Hot* i *StandBy* strana sistema. *Hot* strana je aktivna strana i komponente koja se tu nalaza aktivno učestvuju u razmeni podataka. *StandBy* strana je strana koja prima replikacione podatke i uvek je u stanju pripravnosti da nastavi sa radom ukoliko dođe do problema na *Hot* strani.

4.2 Implementacija realizovanog rešenja

4.2.1 Proizvođač (*Producer*)

U okviru rešenja predstavljenog u ovom poglavlju proizvođač predstavlja glavni izvor podataka. Sama komponenta implementira interfejs *IProducer*. Pomenuti interfejs sadrži dve metode:

- *PublishSync* – Pomoću ove metode se vrši sinhrona objava podataka i dobija se odgovor u obliku *NotifyStatus-a* koji nam govori da li su podaci uspešno isporučeni ili ne.
- *PublishAsync* – Asinhrona objava podataka. Ne čeka se odgovor i nakon poziva metode se nastavlja sa obavljanjem narednih funkcionalnosti.

4.2.2 Potrošač (*Consumer*)

Funkcionalnost ove komponente jeste da šalje zahteve posredniku kada je spremna da obrađuje iste. Potrošač ima zadatak da vodi računa o *Offset-u*, to jest da zna do koje pozicije u particiji je stigao sa zahtevima. Potrošač ima mogućnost da uputi zahtev za podacima na dva načina:

- *SingleRequest* – ovaj zahtev se šalje kada je potrebno dobaviti samo jedan podatak
- *MultipleRequest* – ovaj zahtev se šalje kada je potrebno dobaviti više podataka od jednom

4.2.3 Menadžer (*Manager*)

Menadžer komponenta predstavlja jednu vrstu mehanizma za isporuku podataka. Podaci primljeni od proizvođača se u zavisnosti od vrste slanja (sinhrono ili asinhrono) skladište u red za asinhrono slanje ili se direktno prosleđuju posredniku. Sa jedne strane menadžer ima podignut servis, prema kome proizvođač kreira komunikacioni kanal. Sa druge strane menadžer kreira komunikacioni kanal prema servisu koji je podignut u sklopu posrednika. Pomoću tog kanala menadžer isporučuje prethodno dobijene podatke.

4.2.4 Posrednik (*Broker*)

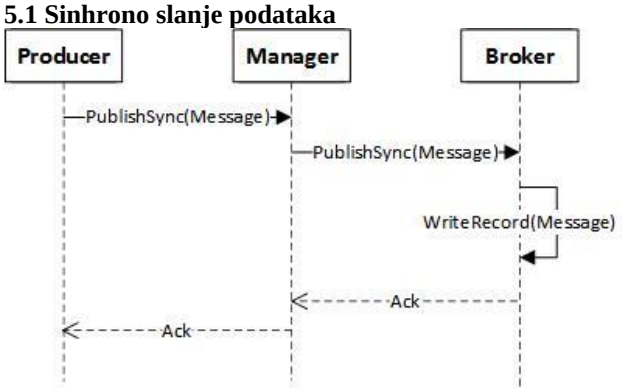
Posrednik predstavlja centralnu komponentu implementiranog rešenja. Sa jedne strane prima podatke od menadžera i skladišti iste dok sa druge strane obrađuju zahteve od potrošača i prosleđuje im tražene podatke. Ima mogućnost rada sa više menadžera i potrošača istovremeno. Posrednik implementira interfejs *IBroke*. *IBroker* nasleđuje sledeće interfejse: *IProducer* i *IConsumer*.

4.2.5 Replikacioni servis

Replikacioni servis čuva *Callback* metodu klijenta kako bi mogao da mu isporuči repliku podataka ako je u *StandBy* stanju, ili kako bi mogao da mu prosledi zahtev za poravnanjem podataka ako je u *Hot* stanju. Takođe čuva *callback* partnerskog replikacionog servisa kako bi mogao da mu prosledi repliku ili podatke na zahtev za poravnanje podataka ako je u *Hot* stanju. Ako je replikacioni servis u *StandBy* stanju tada se kreira komunikacioni kanal prema partneru koji je u *Hot* stanju kako bi mogao da se registruje ili da zatraži zahtev za poravnanjem podataka.

5. DIJAGRAM SEKVENCE

5.1 Sinhrono slanje podataka



Slika 5.1 - Sinhrono slanje podataka

Sinhrono slanje podataka (Slika 5.1) se izvršava na sledeći način:

- Proizvođač podatke šalje menadžeru pomoću metode *PublishSync* i čeka na potvrdu da li su se podaci uspešno isporučili.
- Menadžer dobijene podatke prosleđuje posredniku i takođe čeka potvrdu.
- Posrednik dobijene podatke upisuje u odgovarajuću bazu podataka i kao odgovor vraća menadžeru da li je operacija uspešno izvršena.
- Menadžer tu potvrdu prosleđuje dalje do proizvođaču koji može da nastavi sa radom i u zavisnosti od odgovora vrši izbor daljih koraka.

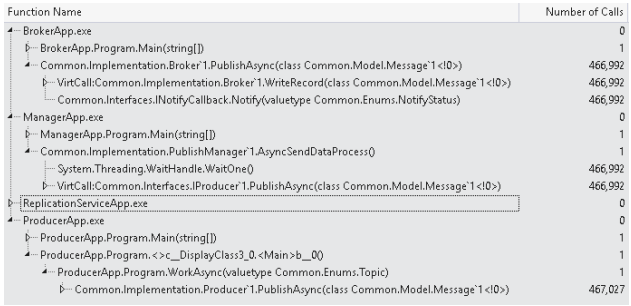
6. PERFORMANSE

6.1 Performanse kod sinhronog slanja podataka

Slika 6.1. Performanse kod sinhronog slanja podataka

Slika 6.1 prikazuje propusnost (broj poruka koje se pošalju u određenom vremenskom intervalu) kod sinhronog slanja podataka. Vremenski period u okviru kog se merila propusna moć jeste jedan minut. Za jedna minut je poslato 64,336 poruka, što se može videti u *Number of Calls* koloni. Za sinhrono slanje poruka poziva se *PublishSync* metoda kod proizvođača. Kod posrednika se poziva metoda *WriteRecord* u okviru koje se čuvaju podaci. Test pokazuje da je i ona pozvana 64,336, što znači da su svi podaci uspešno isporučeni.

6.2 Performanse kod asinhronog slanja podataka



Slika 6.2 - Performanse kod asinhronog slanja podataka

Propusnost asinhronog slanja podataka je očekivano veća nego propusnost kod sinhronog slanja. Vremenski period za koji se merila propusnost je takođe jedan minut. Za asinhrono slanje podataka se koristi funkcija *PublishAsync* kod proizvođača. Sa slike (Slika 6.2) možemo videti da je ta funkcija pozvana 466,992 put. Kod posrednika je funkcija *WriteRecord* pozvana takođe 466,992 puta, što pokazuje da su svi podaci uspešno isporučeni. Za isti vremenski period se prilikom asinhronog slanja pošalje oko sedam puta više podataka nego kod sinhronog slanja.

7. ZAKLJUČAK

Razmena podataka u distribuiranim sistemima predstavlja osnovni zahtev, jer se sa povećanjem broja čvorova povećava i broj zahteva koje treba obraditi. Teži se za rešenjem koje će davati dobre performanse u procesu razmene podataka.

Dalji razvoj rešenja bi mogao da teče u sledeća dva pravca. Prvo bi se mogla omogućiti replikaciju podataka na više partnerskih komponenti istovremeno, čime bi se sigurnost podataka podigla na još viši nivo. Podaci bi bili replicirani na nekoliko geografski udaljenih lokacija.
U drugom pravcu bi mogla da se implementiralogika za prelazak sa *StandBy* na *Hot* i nastavak nesmetanog rada sistema (*Failover*). Ukoliko komponenta koja se nalazi u *Hot* stanju padne i postane neaktivna, komponenta koja se nalazi u *StandBy* stanju treba da preuzme zadatke i nastavi sa izvršavanjem zahteva tamo gde je stala komponenta koja je prethodno bila u *Hot* stanju.

8. LITERATURA

[1] Andrew S. Tanenbaum, *Distributed Systems Principles and Paradigms*
[2] Wikipedia, *Apache Kafka*, <https://en.wikipedia.org/wiki/Apache_Kafka>
[3] The New Stack, *Apache Kafka: A Primer*, <https://thenewstack.io/apache-kafka-primer/>
[4] Kafka Apache, *Introduction*, <https://kafka.apache.org/intro.html>
[5] Microsoft, *What is .NET framework*, <https://dotnet.microsoft.com/learn/dotnet/what-is-dotnet-framework>
[6] Tech Target, *C Sharp*, <https://searchwindevelopment.techtarget.com/definition/C>
[7] Tech Target, *Failover*, <https://searchstorage.techtarget.com/definition/failover>
[8] Wikipedia, *Windows Communication Foundation*, <https://sr.wikipedia.org/sr-el/Windows_Communication_Foundation>
[9] Tutorials Teacher, *Generics in C#*, <https://www.tutorialsteacher.com/csharp/csharp-generics>
[10] Microsoft, *Generics (C# Programming Guide)*, <https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/generics/>

Kratka biografija:

Kristijan Salaji rođen je 02.01.1995. godine u Novom Sadu. Završio je srednju elektrotehničku školu "9. Maj" 2014. godine u Bačkoj Palanci. Iste godine je upisao osnovne akademske studije na Fakultetu Tehničkih Nauka u Novom Sadu, smer Elektroenergetski Softverski Inženjering, koje je završio 2018. Odmah za tim upisuje master akademske studije na Fakultetu Tehničkih Nauka, smer Primenjeno Softversko Inženjerstvo.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004:007.5
DOI: https://doi.org/10.24867/07BE42Panic

FUZZY KONTROLER ZA UPRAVLJANJE ENERGIJOM U PAMETNOJ KUĆI

FUZZY ENERGY MANAGEMENT CONTROLLER FOR SMART HOME

Stefan Panić, *Fakultet tehničkih nauka, Novi Sad*

Oblast: ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj: *Rad se bazira na analiziranju potrošnje električne energije koji se najviše oslikavaju kroz upotrebu HVAC sistema i osvetljenja*
Ključne reči: *Fuzzy, HVAC, osvetljenje, pametna kuća*
Abstract: *The study is based on analyzing the electricity consumption that is most reflected through the use of HVAC systems and lighting*
Keywords: *Fuzzy, HVAC, lighting, smart house*

1. UVOD

Pametna kuća se može definisati kao ona koja je opremljena pametnim uređajima [1]. Umreženost uređaja u kući omogućava transfer informacija između istih, dok gateway povezuje pametne uređaje sa internetom. Gateway omogućava uređajima u kući da se konektuju na internet i preuzmu nove usluge. Provajderi usluga su zaduženi za nove usluge stanovnika i njihov pristup. Pametni uređaji mogu da komuniciraju sa korisnicima, pa čak i da ih osmatraju. Moderna tehnologija se koristi da bi se konstruisao ambijent u kojem su mnogi uređaji povezani i automatizovana. Pametna kuća pruža mogućnost za poboljšanje kvaliteta korisnikovog života tako što će stvoriti fleksibilnu, komfornu, zdravu i efikasnu sredinu. Sistemi pametne kuće su integrisani pametnim uređajima i poboljšanim uslugama koji se tiču okoline, bezbednosti, zabave, komunikacije, pomoći zdravstvene nege i električnih kućnih uređaja.

2. PAMETNI KUĆNI SISTEMI ZA UPRAVLJANJE ENERGIJOM

EPRI (Institut Electrical Power Research) je 1998. godine, sproveo istraživanje o kompleksnim interaktivnim mrežnim sistema kako bi se razvila visokopouzdana i u potpunosti automatizovana mreža u SAD, što predstavlja prototip američke pametne mreže. Predložen je termin Intelli-Grid, 2002. godine, i taj koncept je široko prihvaćen za indikaciju budućeg razvijanja električnih mreža. U Evropi je 2005. godine razvijena Platforma za tehnologije pametne mreže, a zatim i sprovedeno istraživanje, 2006. godine, kako bi se formulisao koncept i okvir evropske pametne mreže. Nešto kasnije, Department of Energy u SAD (United States) je objavio izveštaj o pametnim mrežama ("The Smart Grid"), u decembru 2007. godine, i on integriše evropske ideje i koncepte u američku pametnu mrežu kako bi se stvorio pouzdaniji i izdržljiviji izvor obnovljive energije [2]. Stvaranje modernizovane infrastrukture za pametne gradove postao je prioritet u mnogim zemljama zbog brojnih prednosti koje može da pruži. Korišćenje pametne tehnologije ima važnu ulogu u poboljšanju izdržljivosti i očuvanja energije za korisnike, a takođe utiče i na šablon dnevne potrošnje energije. Skorašnji napredak informacionih i komunikacijskih tehnologija, poput AMI (advanced metering infrastructure), tehnologije pametnih senzora, birdrectional komunikacije, pametnih kućnih uređaja , kućne mreže HAN (home area network) i sistema skladištenja energije u kući HESS (home energy storage system) je primenjen.

Ovaj rastući trend omogućava tehničke temelje infrastrukture pametnih kuća sa sistemima za kontrolu energije HEMS (home energy management system).

Pametni HEMS je neophodan za uspešno upravljanje pametnim mrežama. On prati brojne kućne uređaje u odnosu na korisnikove navike putem interfejsa u pametnim kućama, kako bi se smanjili troškovi energije i poboljšala efikasnost uređaja. Rastom svesti o bezbednosti globalne energije, sve više se distribuiraju obnovljivi izvori, poput vetrenjača, solarnih panela, električnih vozila PEV (plug-in electric vehicle) i u njih se integriše mreža kako bi se postigla aktivna distribucija.

Zajedno sa ubrzanim i naprednim električnim uređajima i alternativnim energetskim tehnologijama, izgradnja objekata koji koriste obnovljive izvore energije se može pripojiti pametnom HEMS-u, kako bi se poboljšala efikasna konverzija i iskorišćavanje energije [3]. Ovo vodi ka fundamentalnom prenosu ka modernim sistemima za kontrolisanje energije i cyber-physical HEMS-u od tradicionalnih centralizovanih infrastruktura, za veće geografske regione obnovljivih energetskih izvora putem pametnih energetskih sistema. Informacijama koje putuju u dva smera, između provajdera energije i korisnika pametne mreže, masovni HEMS učestvuje u mehanizmu zahteva potražnje za očuvanje energije i kooperacije [4].

Zahtev potražnje predstavlja promene u korišćenju električne energije od strane korisnika sa uobičajenih šablona potrošnje, usled promena u ceni električne energije i podstiče korisnike da smanje potrošnju energije tokom perioda viših tarifa ili kada je pouzdanost izvora ugrožena.

Korisnici HEMS-a mogu da promene zahteve potrošnje uređaja ili automatski ili ručno na niže tarife kako bi troškovi električne energije bili niži. U tipičnoj pametnoj kući, uređaji kontrolisani termostatom, poput grejanja, ventilacije, klima uređaja, bojlera i frižidera, troše najviše struje.

Konstantno rastuća potrošnja uređaja i energetska kriza čine upotrebu HEMS-a primamljivom korisnicima.

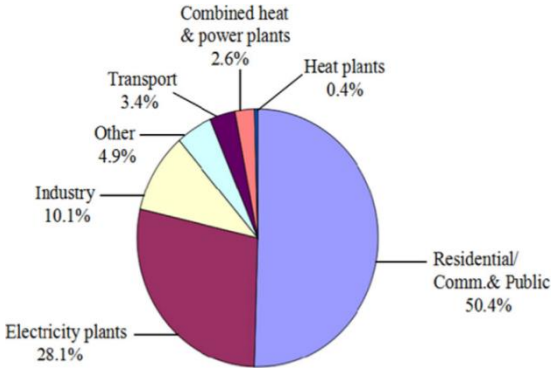
NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Velimir Čongradac, vanr. prof.

3. OBNOVLJIVI IZVORI ENERGIJE U PAMETNIM KUĆAMA

Od 1990. godine, upotreba obnovljivih energetskih izvora je porasla prosečnom godišnjom stopom od 2% [5]. Obnovljiva energija se koristi u brojnim poljima, kao što su industrijski, stanarski, komercijalni i javni sektor.

Samo 31,1% (Slika 1.) obnovljive energije se koristi za proizvodnju struje i toplote, dok je 50% iskorišćeno u stanarske, komercijalne i javne svrhe. Istraživanja na temu obnovljivih izvora energije u HEMS-u ima velik značaj i razvojni prospekt. Ubrzanim razvojem tehnologije održive energije i rastućeg zahteva za generaciju niske emisije, upotreba obnovljive energije ima obećavajuću perspektivu kada je reč o pametnim kućama. Sa tehničkih i ekonomskih aspekata, izvodljivo je zameniti fosilna goriva obnovljivom energijom. Daljim razvojem tehnologije pametne mreže, koje uključuju komunikaciju i monitoring, kontrolu i self-healing, energija pametne kuće je unapređena prilagođavanju obnovljivim izvorima.

U Nemačkoj trenutno, oko 0,9% stanovništva koristi solarne panele. Stanovnici prodaju energiju tokom dana po visokoj ceni, a noću je kupuju po niskoj ceni. Zahvaljući lakoj instalaciji i malim troškovima, solarni PV su u velikoj meri korišćeni u pametnim kućama. Kipar, kao vodeći po korišćenju solarnih grejača vode na svetu, je zemlja gde 92% domaćinstava i 53% hotela koristi ove solarne sisteme. Produktivnost solarnih PV ćelija u Kini 2008. godine je oko 937,363 m2 solarnih kolektora, što je približno 1m2 po osobi. Produktivnost solarnih PV ćelija u Kini u 2008. godini je bila oko 4GW, od čega su 3GW kapaciteti modula. Kumulativni kapacitet instaliranih PV je snage od 150MW. U isto vreme solarni grejači vode pokrivaju oko 125 miliona m2 u građevinama, što dostiže oko 60% svetske količine.



Slika 1. Svetska potrošnja obnovljivih izvora energije u 2012. godini

4. FUZZY KONTROLER ZA UPRAVLJANJE ENERGIJOM

Pametna elektroenergetska mreža je proistekla od tradicionalne elektroenergetske mreže. Da bi se ovi parametri postigli, nekoliko vrsta uređaja moraju biti povezani. Takođe senzori, softveri i automatski daljinski upravljači uređaja su uključeni u ovu mrežu. Komunikacija ima važnu ulogu kada je reč o pametnoj elektroenergetskoj mreži. Ona se vrši između potrošača i uređaja.

Podaci se šalju i primaju kod obe strane zbog optimizacije energije i održivosti mreže. Važna osobina pametne mreže je i DSM(demand side management). On optimalno zakazuje uredjaj kako bi se troškovi električne energije smanjili i kako bi se postigla najefikasnija upotreba električne energije [6]. DSM ima dva aspekta: kontrolu uredjaja i na zahtev potražnje DR (demand response).

Drugo navedeno se fokusira na efikasan menadžment energije da bi se smanjile potencijalne opasnosti ili nestanak struje.Ima značajnu ulogu u smanjenju odnosa maksimalne i prosečne potrošnje PAR-a (peak to average ratio), troškova elektične energije i potrošnje elektrčne energije. Takođe pomaže pouzdanosti i održivosti mreže.

5. SIMULACIJA REZULTATA I DISKUSIJA

Ovo poglavlje opisuje rezultate simulacija upravljača za kontrolu energije. Korišćena je fuzzy logika i heurističke tehnike kako bi se postigli željeni ciljevi.

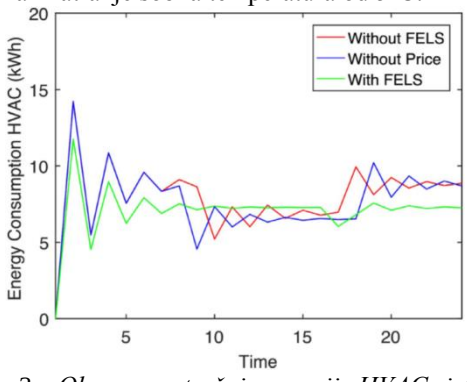
U ovom istraživanju, fuzzy logika nema stroga pravila za selektovanje članova funkcija.[7]. Tačnost oba fuzzy kontrolera je zagarantovana. Kako bi se potvrdio EMC sistem, predstavljena su tri istraživanja. U prva dva, govori se o uticaju koji fuzzy HVAC i kontroler osvetljenja imaju na potrošnju energije.

Troškovi električne energije, PAR i njihov uticaj na korisnikovu udobnost su predstavljeni u trećem istraživanju. U ovom istraživanju se takođe porede i heurističke tehnike.

6. ISTRAŽIVANJA FUZZY HVAC KONTROLERA

Ovo istraživanje se bavi fuzzy HVAC kontrolerom. Postoje dva režima HVAC sistema koja su testirana: manuelni režim i autonomni režim. U manualnom režimu, postavne tačke temperature su podešene ručno od strane korisnika. Nasuprot tome, fuzzy HVAC kontroler je korišćen za generisanje postavnih tačaka temperature u autonomnom režimu, koji je poznat i kao FELS.

Parametri ulaza ovog kontrolera su: spoljašnja temperatura, obaveze korisnika, zahtev energije i cena električne energije. Istraživani su i efekti ovih parametara na utrošak energije. Slika 2. pokazuje potrošnju energije HVAC sistema za 24h. Može se uočiti da je potrošnja energije bez FELS-a veća nego sa njim. Upoređena je i potrošnja energije FELS-a sa režimom omogućene cene bez njega. Rezultati pokazuju da je potrošnja energije veća nego FELS sa cenom. Potrošnja energije HVAC sistema u sva tri slučaja ista do 7h. Posle 7h, cena energije i potrebna energija za rad uređaja se menjaju. Parametri obaveza korisnika se takođe menjaju jer korisnik nije prisutan od 8 do 15h, tako da šablon konzumiranja energije varira između 7 i 22h. U 1h, potrošnja energije je visoka jer je uzeta u razmatranje sobna temperatura od 5◦C.



Slika 2 – Obrazac potrošnje energije HVAC sistema

7. FUZZY SISTEM ZA UPRAVLJANJE ENERGIJOM U PAMETNIM MREŽAMA

HVAC sistemi približno čine 64% i 57% kompletne potrošnje energije stanara u Kanadi i SADu [8]. HVAC sistemi u domovima čine jedno od glavnih potrošača u periodu kada je potražnja struje najveća. Sa druge strane, jedan od glavnih ciljeva inicijative za pametnu mrežu je poboljšanje uvida u smanjenje mrežne voltaže kao i omogućavanje korisnikovog udela u operacije sistema, naročito putem pametnih brojila [9], kontrolisanja sistema pametne energije i pametnih domova [10]. Sa određenim poboljšanjima u komunikaciji mreža i širenju razvijenih pametnih brojila, problemi kontrolisanja vrhunca učitavanja se prebacuju do strane korisnika [11].

Sproveden je holistički pregled kako bi se sumirale inicijative i objekti koji imaju mogućnost da pomognu stanarima da potencijalno uštede energiju. Uređaji koji prikazuju energiju, tako što daju povratne informacije korisnicima o potrošnji energije, mogu značajno da doprinesu smanjenju konzumiranja energije, tako što će prebaciti zahteve za energijom na vreme niže tarife. Predlog je upotreba pametnih brojila i uređaja, koji mogu da podstaknu korisnike da u budućnosti imaju aktivnu ulogu u pametnim energetskim mrežama..

U svakoj vezi komunikacije unutar pametne mreže, razmatranje bezbednosnih pitanja je veoma važno.Pametna brojila, kao tehnologija koja se deli između korisnika i pametnih mreža, može da omogući stanarima da postanu osnovni deo sistema električne energije. Cene struje koje variraju, kao što su TOU stope, cene u stvarnom vremenu i kombinacije ovih mehanizama, omogućuju različite prilike za stanare da smanje konzumiranje energije i troškove struje, tako što će prebaciti režime rada sa više na nižu tarifu [12]. Strategije kontrole učitavanja HVAC sistema mogu da se izvrše raspodelom učitavanja po odgovoru na različite parametre, kao što su varirajuće cene [13], varijacije u temperaturi [14] i obaveze korisnika [15].

Uloga upotrebe senzora pisutnosti u pametnim mrežama, za očuvanje energije smanjenjem postavnih tačaka temerature HVAC sistema, kada u kući stanari nisu prisutni može mnogo uticati na uštedu. Opcije tehnologije, kao što su iskorišćavanje kuće, su sve mreže [16] i instaliranje uređaja praćenje HVAC potošnje energije [17] kao i termostati koji se mogu programirati [18] su takođe dostupni za asistiranje stanarima kako bi se kontrolisala i smanjila upotreba energije raspodelom zahteva kućnih uređaja i HVAC sistema tokom viših tarifa naplaćivanja struje.

Komunikacijski termostati koji se mogu programirati (PCT), termostati koji odgovaraju na cene [19] i termostati koji primećuju prisutnost [20] su korišćeni za automatsko kontrolisanje HVAC sistema, gde korisnici unose svoje svakodnevne obaveze (tj. vremenske intervale) i potrebe (tj. podešavanje temperature). Termostati koji odgovaraju na cene i PCT potencijalno imaju mogućnost za dvostranu komunikaciju, kao što je korišćenje ZigBee komunikacijskog protokola (IEEE 802.15.4) sa uređajima kroz razvijena pametna brojila, kako bi učestvovali u programima za zahtev potražnje (DR), po izboru korisnika [21].

Ovi termostati mogu da prime signale naplaćivanja od pametne mreže i da automatski povećaju ili smanje početna podešavanja do nivoa prethodno postavljenog od strane korisnika. Termostati koji primećuju prisutnost takođe prate i automatski menjaju podešavanja kada u prostoru/sobi nema ljudi. Međutim, postoje brojni nedostaci kod ovakvih termostata. Utvrdjeno je da čak i postojeći uređaji za kontrolisanje energije ne mogu uvek da uštede energiju zbog zavisnosti od korisnikovog angažovanja. U ovom slučaju, korisnici konstantno moraju da ponovo podešavaju predefinisane postavke kako bi se održao termalni komfor. Iz toga sledi da korisnicima stvara neugodnost konstantno menjanje podešavanja postavnih vrednosti usled varijacija u ceni ili prisutnosti.

Može da se desi i da korisnici zaborave, zanemare ili čak i odustanu od podešavanja termostata kada im je termalna udobnost ugrožena zbog prisutnosti DR programa. Glavni razlog toga jeste nedostatak informacija i prilagođavanja korisnikovim potrebama, kada je reč o toploti u postojećim PCT ili termostatima koji primećuju prisutnost.

Razvijanje autonomnih fuzzy tehnika za buduće pametne termostate može pomoći korisnicima da učestvuju u DR programima bez ikakve interakcije sa termostatima, kako bi se očuvala energija i umanjili troškovi, a zadržala udobnost. Sinergija modela prediktivne kontrole i predviđanja vremenske prognoze, stvoreni su da bi se poboljšao učinak energije u objektima, a zadržao termalni komfor.

8. MODEL ADAPTIVNOG FUZZY LOGIČKOG SISTEMA

Automatizovanje rada kućnih uređaja nije dugoročno rešenje zbog njihove aktivnosti i korišćenja šablona koji zavise od faktora poput vremenskih uslova i cene električne energije. Stoga, kao dodatak stvaranju autonomnih sistema, mora da se pronađe i rešenje za adaptaciju promenama u korisnikovom ponašanju koje se mogu vremenom stvoriti. Ovo nas vodi kroz problem sa stanovišta sistema; interakcija određenih podsistema, gde svaki ima sopstvene osobine.

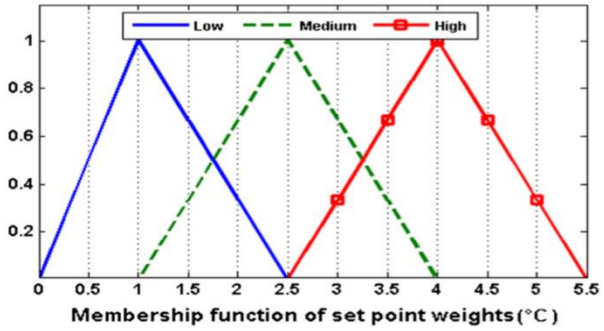
Na ovaj način se održava generalnost sistema, tako da razvijeni adaptivni autonomni sistemi, kao što je pomenuti termostat, mogu da se primene na bilo koji objekat.

9. FUZZY DONOŠENJE ODLUKA ZA ADAPTACIJU

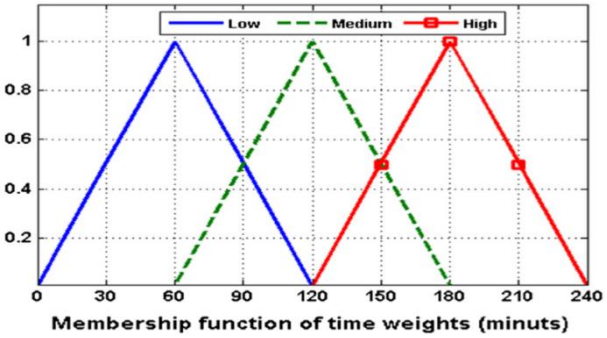
Ova funkcija se dodeljuje svakoj težini vektora učenja ukoliko se detektuje ikakva promena u vektorima učenja. Slika 3 i 4 pokazuju funkcije težina povezane sa svakim elementom vektora učenja. Razlozi za definisanje trouglastih MF su bazirani na nekoliko pretpostavki. Navike korisnika se predviđaju nakon 3 uzastopne promene. Dodeljivanje drugih vrsta MF, poput trapezastog, nemaju isti efekat na tačnost predviđenih vrednosti dok odabir drugih MF povećava broj pravila koja čine sistem novčano skupljim. Ove težine se dodeljuju svakom elementu adaptivnih i vektora učenja za bilo koji deo dana koji je pod osmatranjem, u odnosu na njihove promene sa početnih vrednosti.u ovim grafikonima, težina High označava najveću promenu sa početne vrednost, a Low označava manju promenu sa tipične vrednosti.

Korisnik može da odluči da li će promeniti te vrednosti u bilo koje doba dana ukoliko je nezadovoljan trenutnom unutrašnjom temperaturom. U ovom slučaju sistem mora da detektuje i razmotri promenjene vrednosti da bi se zamenile kontrole koje mogu predstavljati korisnikove nove potrebe. HVAC kontroler (termostat) proverava

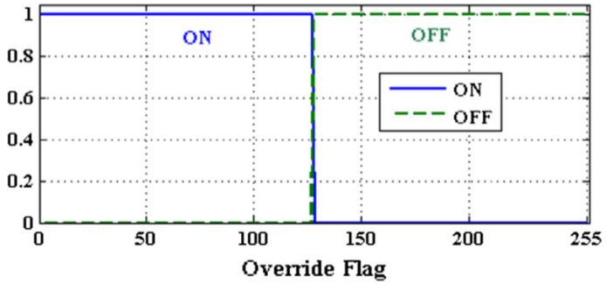
trenutno stanje postavki i daje korisniku promenjene vrednosti. Samo jedna funkcija može biti primljena u određeno vreme. Slučaj kada je override flag uključen, znači da je odluka doneta SFLL načinjena od strane korisnika. Nove odluke moraju biti snimljene za buduće adaptacije u zavisnosti od doba dana. Ukoliko je override flag isključen, to znači da je sistem i dalje u svom normalnom režimu (Slika 5.).



Slika 3 – Funkcije članstva povezane sa prebacivanjem zadatih vrednosti kao sistemski ulaz



Slika 4 – Funkcije članstva povezana sa pomeranjem početnog vremena i završnog vremena kao ulaza sistema



Slika 5 – Funkcije članstva override flag-a.

10. SIMULACIJA REZULTATA I UČINAK RAZVIJENOG ALGORITMA

Da bi se potrvdio učinak razvijenog algoritma koji štedi energiju i funkcionalan je pri adaptiranju promenama korisnikovih potreba, urađene su simulacije u nekoliko različitih scenarija. AFLM pristup je proverio i manuelni i autonomni režim.

11. ZAKLJUČAK

Korišćenjem veštačke inteligencije, uključujući FL, neuronske mreže i evoluciono računanje, omogućava implementaciju moćnijih, kompleksnijih kontrolnih sistema i sistema za donošenje odluka. Hibridni pametni kontrolni sistem je dostojno rešenje kada je model okoline toliko kompleksan da bi matematički model bio nelinearan ili kada bi ga bilo nemoguće razviti i stvoriti. Stoga, hibridni pametni kontrolni sistemi za generisanje kontrolnih pravila dobijenih iz primera moraju dalje i detaljnije biti proučavani.

12. LITERATURA

[1] V. Ricquebourg, D. Menga, D. Durand, B. Marhic, L. Delahoche, C. LogeThe smart home concept: our immediate future, 2006 IEEE International Conference on E-Learning in Industrial Electronics;(2006), pp. 23-28

[2] Obama. Remarks by the president on recovery act funding for smart grid technology; 2009.

[3] Li C, Shi H, Cao Y, Wang J, Kuang Y, Tan Y. Comprehensive review of renewable energy curtailment and avoidance: a specific example in China. Renew Sustain Energy Rev 2015;41:1067–79.

[4] Zhou S, Wu Z, Li J, Zhang X. Real-time energy control approach for smart home energy management system. Electr Power Compon Syst 2014;42:315–26.

[5] IEA. Renewables information 2014. International Energy Agency 2014.

[6] A. Ahmad, N. Javaid, N. Alrajeh, Z.A. Khan, U. Qasim, A. Khan, A modified feature selection and artificial neural network-based day-ahead load forecasting model for a smart grid, Appl. Sci. 5 (4) (2015) 1756–1772.

[7] Z. Zhao, W.C. Lee, Y. Shin, K.-B. Song, An optimal power scheduling method for demand response in home energy management system, IEEE Trans. Smart Grid 4 (3) (2013) 1391–1400. 118 R. Khalid et al. / Sustainable Computing: Informatics and Systems 21 (2019) 103–118

[8] Household energy consumption and expenditures, consumption by end use. Energy Information Administration (EIA); 2010.

[9] U.S. Department of Energy. Smart grid system report; July 2009.

[10] Vojdani A. Smart integration. IEEE Power Energy Mag 2008;6(6):71–9.

[11] Yan C et al. A novel air-conditioning system for proactive power demand response to smart grid. Energy Convers Manage 2015;102:239–46.

[12] Gangale F, Mengolini A, Onyeji I. Consumer engagement: an insight from smart grid projects in Europe. Energy Policy 2013;60:621–8.

[13] Faruqui A, Sergici S. Household response to dynamic pricing of electricity: a survey of 15 experiments. J Regul Econ 2010;38(2):193–225.

[14] Karjalainen S. Thermal comfort and use of thermostats in Finnish homes and offices. Build Environ 2010;44(6):1237–45.

[15] Johnson BJ et al. A method for modeling household occupant behavior to simulate residential energy consumption. In: Innovative smart grid technologies conference (ISGT), 2014 IEEE PES. IEEE; 2014.

[16] Koomey J, Brown RE. The role of building technologies in reducing and controlling peak electricity demand. Report number, LBNL-49947; 2010.

[17] Vakiloroaya V et al. A review of different strategies for HVAC energy saving. Energy Convers Manage 2014;77:738–54.

[18] Consumer reports. Programmable thermostats lab test - some make saving easier; 2010.

[19] Chanana S, Arora M. Demand response from residential air conditioning load using a programmable communication thermostat. Int J Electr Comput Energetic Eng 2013;7(12).

[20] Woolley Jonathan, Pritoni Marco, Modera Mark. Why occupancy-responsive adaptive thermostats do not always save and the limits for when they should. In: ACEEE summer study on energy efficiency in buildings. p. 337–50.

[21] Saputro N, Akkaya K, Uludag S. A survey of routing protocols for smart grid communications. Comput Netw 2012;56(11):2742–71.

Kratka biografija

Stefan Panić rodjen je 16.08.1991. godine u Sremskoj Mitrovici. Osnovnu školu „Boško Palkovljević Pinki“ završio je u Sremskoj Mitrovici, Mitrovačku gimaziju, smer prirodnomatematički završava 2010. godine. Iste godine upisuje Fakultet Tehničkih Nauka u Novom Sadu, oblast Elektrotehnika i računarstvo, smer Računarstvo i automatika. Kasnije se usmerava na Automatiku i upravljanje sistemima gde je diplomirao 2015. godine i zatim upisuje master akademske studije na Fakultetu Tehničkih Nauka.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.3
DOI: https://doi.org/10.24867/07BE43Cicmil

PROGRAMSKA PODRŠKA ZA ANALIZU NUMERIČKIH METODA ZA MERENJE EFEKTIVNE VREDNOSTI NAPONA

SOFTWARE FOR ANALYSIS OF NUMERICAL METHODS FOR MEASURING RMS VOLTAGE

Miloš Cicmil, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je opisana projektovana programska podrška za analizu 6 različitih metoda za aproksimaciju efektivne vrednosti (Root mean square, RMS) napona naizmeničnog signala (koji ne mora biti prostoperiodičan), na osnovu semplovanih odbiraka, redom pravougaone, trapezne i Simpsonovih metoda (polinoma drugog i trećeg), kao i dve modifikovane Simpsonove metode [1].*

Ključne reči: *Merenje napona, Efektivna vrednost napona, Root mean square, RMS, Numerička integracija, Pravougaona metoda, Trapezna metoda, Simpsonova metoda, Numerička analiza*

Abstract – *This paper describes the operation of the developed software for analysis of 6 different methods for approximation of root mean square (RMS) voltage of a sampled signal, respectively rectangular, trapezoidal and original Simpson's polynomial methods, as well as two modified Simpson's methods.*

Keywords: *Voltage measurement, Root mean square, RMS, Numerical integration, Rectangle method, Trapezoidal method, Simpson method, Numerical analysis*

1. UVOD

Efektivna vrednost napona naizmeničnog signala je definisana kao kvadratni koren iz srednje kvadratne vrednosti (Root mean square, RMS) funkcije napona od vremena, *u(t)*, po formuli:

$$U_{eff} = \sqrt{\frac{1}{T} \cdot \int_0^T u^2(t)\,dt}. \qquad (1)$$

gde je *T* trajanje jedne periode signala. Ova vrednost je od izuzetne važnosti u elektrotehnici (njoj je npr. direktno srazmerna snaga signala) i samim tim je veoma bitno njeno efikasno merenje.

S obzirom da se savremena tehinka za merenje uglavnom zasniva na merenjima trenutnih, diskretnih vrednosti, za kontinualne signale nepoznate prirode (nepoznate zavisnosti od vremena) je stoga nemoguće tačno izračunati RMS, tj. efektivnu vrednost, te se za njen proračun koriste različite metode čiji je cilj minimiziranje nastale greške, uz što manju "cenu" po pitanju računarske zahtevnosti, koja je uglavnom srazmerna količini i kompleksnosti matematičkih operacija preko kojih je metoda definisana.

U radu je prikazana programska podrška, namenjena za analizu šest metoda za aproksimaciju, tj. proračun RMS različitih naizmeničnih signala (zadatog preko prvih deset harmonika). Analizira se pre svega zavisnost njihove greške od frekvencije semplovanja (*fs*), tj. odnosa *fs* i frekvencije merenog signala *f*, i zadatog naponskog "praga" koji definiše početak i kraj merenja (čije funckionisanje je objašnjeno u daljem tekstu).

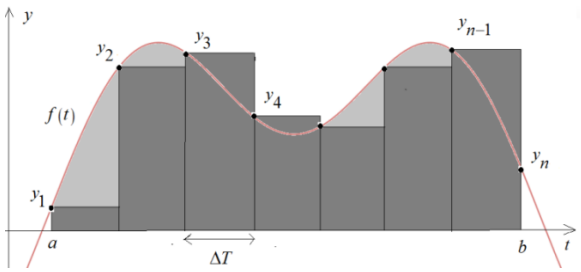
2. ANALIZIRANE METODE ZA PRORAČUN RMS

Ispitivane metode su u suštini metode za numeričku aproksimaciju integrala iz formule (1), za ekvidistantne odbirke (semplove) nastale usled konstantne *fs*, i one su redom:

1. (Leva) pravougaona metoda
2. Trapezna metoda
3. Prva Simpsonova metoda (metoda polinoma drugog reda)
4. Druga Simpsonova metoda (metoda polinoma trećeg reda)
5. Modifikovana Simpsonova metoda polinoma drugog reda
6. Modifikovana Simpsonova metoda polinoma trećeg reda

2.1. Pravougaona metoda

Jedna od najčešće korišćenih integracionih metoda za numeričku integraciju jeste pravilo levih ili desnih pravougaonika [1]. Ovde je analizirana samo metoda levih pravougaonika i njena ilustracija je data na Slici 1.



Slika 1. *Ilustracija leve pravougaone metode [2]*

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

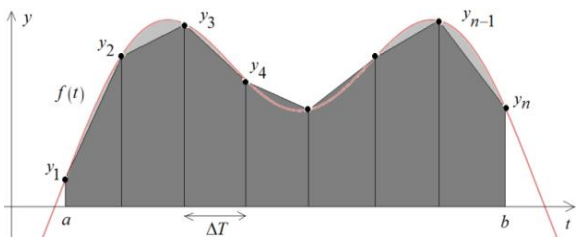
Integral se aproksimira sumom površina pravougaonika gde se za jednu njegovu stranicu uzima vrednost širine podintervala ΔT, a za vrednost druge stranice pravougaonika vrednost funkcije u levoj krajnjoj tački tog podintervala [1]. Odatle se dobija izraz za aproksimiranu RMS:

$$U_{eff} \approx \sqrt{\frac{1}{T}\sum_{i=1}^{n} u^2(i\Delta T)\Delta T} = \sqrt{\frac{\Delta T}{T}\sum_{i=1}^{n} u^2(i\Delta T)} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} u^2(i\Delta T)} \quad (2)$$

Odbirci $u(i\Delta T)$ se uzimaju redom počevši od prve semplovane vrednosti veće od nule, uzimajući potom sve pozitivne, pa potom sve negativne vrednosti (kraj obeležava prvi potom opet pozitivan odbirak, koji se ne uzima). Takođe, umesto nule, zadati "prag" koji definiše početak i kraj merenja, tj. semplovanja, se u aplikaciji može postaviti na proizvoljnu vrednost, i to je jedna od ključnih veličina čiji uticaj se projektovanim alatom može analizirati.

2.2. Trapezna metoda

Trapezna metoda numeričke integracije se zasniva na sumiranju integrala podintervala, koji geometrijski predstavljaju trapeze, kao što je prikazano na Slici 2.



Slika 2. *Ilustracija trapezne metode [2]*

Za ovu metodu se radi tačnije aproksimacije uzima jedan više odbirak (sempl) nego za pravougaonu metodu, tj. i prethodno pomenuti prvi odbirak veći od praga (jer se inače dobija veća greška uvek istog znaka (pozitivna sistematska greška) [1]).

Po ovoj metodi se za RMS dobija izraz:

$$U_{eff} \approx \sqrt{\frac{1}{n}\left(\sum_{i=2}^{n} u_i^2 + \frac{u_1^2 + u_{n+1}^2}{2}\right)} \quad (3)$$

gde je n broj odbiraka ne računajući poslednji veći od praga, tj. kao po pravougaonoj metodi.

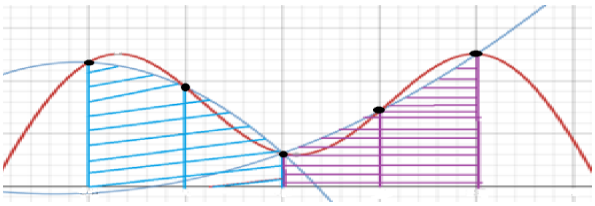
Za razliku od pravougaone metode, u kojoj su svi sabirci istog značaja, ovde za efektivnu vrednost možemo uočiti da se prvi i poslednji odbirak uzimaju sa upola manjom težinom [1].

2.3. Prva Simpsonova metoda (metoda polinoma drugog reda)

Na Slici 3 je ilustrovana numerička integracija funkcije integracionim polinomom drugog reda [1].

Tačna vrednost ovog integrala jednaka je površini ispod date funkcije (crvena kriva), a približna vrednost površini ispod krive polinoma nad periodom (plave krive) [1].

Takođe se i ovde, kao i za trapeznu metodu, radi tačnije aproksimacije, uzima jedan odbirak više nego za pravougaonu.



Slika 3. *Ilustracija prve Simpsonove metode (metode polinoma drugog reda) [1]*

Za tako definisan proračun integrala se za RMS dobija izraz:

$$U_{eff} \approx \sqrt{\frac{1}{n-1}\left(\frac{u_1^2 + u_n^2}{3} + \frac{4}{3}\sum_{i=1}^{\frac{n-1}{2}} u_{2\cdot i}^2 + \frac{2}{3}\sum_{i=1}^{\frac{n-3}{2}} u_{2\cdot i+1}^2\right)}. \quad (4)$$

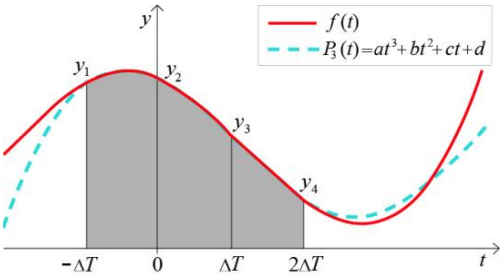
gde je n broj uzetih odbiraka (dakle računajući i poslednji odbirak veći od praga).

Nedostatak ove metode je da je njena primena moguća jedino ukoliko je broj odbiraka neparan.

2.4. Druga Simpsonova metoda (metoda polinoma trećeg reda)

Ova metoda se zasniva na podeli celog intervala integracije na grupe od po četiri uzastopna odbirka i aproksimaciji integrala funkcije nad tim intervalima polinomom trećeg reda. Usled toga ona se može primeniti samo ukoliko je broj odbiraka oblika $3k+1$, gde je k ceo broj (npr. 10, 13). Za nju se takođe, kao za prethodne dve metode, uzima jedan odbirak više nego za pravougaonu metodu, tj. završno sa odbirkom koji pređe prag nakon periode.

Na Slici 4 je ilustrovana njena primena.



Slika 4. *Ilustracija Druge Simpsonove metode (metode polinoma trećeg reda) [2]*

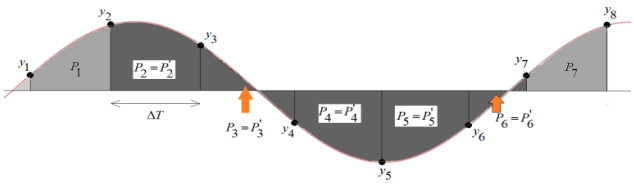
Ovde se za aproksimaciju integrala dobija izraz:

$$U_{eff} \approx \sqrt{\frac{1}{n}\left(\begin{array}{l}\frac{3}{8}\left(u_1^2 + u_n^2\right) + \frac{9}{8}\sum_{k=0}^{n-2} u_{2+3k}^2 \\ + \frac{9}{8}\sum_{k=0}^{n-1} u_{3+3k}^2 + \frac{3}{4}\sum_{k=0}^{n-3} u_{4+3k}^2\end{array}\right)} \quad (5)$$

2.5. Modifikovana Simpsonova metoda polinoma drugog reda

Na Slici 5 je prikazana ilustracija metode koja predstavlja modifikaciju Simpsonovog pravila, koju predlažu autori rada [1]. Ona je zamišljena kao rešenje za problem koji se javlja ako je broj odbiraka paran, i ona se, za razliku od originalne Simpsonove metode polinoma drugog reda, može primeniti na bilo kom broju odbiraka.

Za nju se uzima još jedan dodatni odbirak u odnosu na prethodne tri metode, dakle uključuju se i prva dva odbirka koja pređu prag nakon periode.



Slika 5. *Ilustracija modifikovane Simpsonove metode polinoma drugog reda [2]*

Proračun integrala se vrši sumiranjem integrala nad podintervalima određenim sa svaka 3 uzastopna odbirka, kao što je prikazano na Slici 5, te dolazi do dvostrukog uračunavanja svih podintervala osim prvog i poslednjeg, što se koriguje u konačnom izrazu za vrednost proračunatog integrala, i odatle se za proračunati RMS dobija izraz:

$$U_{eff} \approx \sqrt{\frac{1}{(n-2)}\left(\frac{u_1^2+u_n^2}{6}+\frac{5}{6}\left(u_2^2+u_{n-1}^2\right)+\sum_{i=3}^{n-2}u_i^2\right)}. \quad (6)$$

gde je *n* broj odbiraka uzetih za aproksimaciju.

2.6. Modifikovana Simpsonova metoda polinoma trećeg reda

Ova metoda je zamišljena sa istom idejom kao i prethodna, da sumiranjem integrala nad podintervalima određenim sa svaka četiri uzastopna odbirka, eliminiše potrebu originalne Simpsonove metode polinoma trećeg reda za specifičnim brojem odbiraka, tako da ova metoda može kao i prethodna da se primeni za bilo koji broj odbiraka.

Za ovu metodu se uzima još jedan dodatni odbirak u odnosu na prethodnu metodu, dakle uključuju se i prva tri odbirka koja pređu prag nakon periode, i kod nje dolazi do trostrukog uračunavanja svih podintervala osim spoljašnjih, što se koriguje u konačnom izrazu za vrednost proračunatog integrala, i odatle se za proračunati RMS dobija izraz:

$$U_{eff} \approx \sqrt{\frac{1}{(n-3)}\left(\frac{u_1^2+u_n^2}{8}+\frac{1}{2}\left(u_2^2+u_{n-1}^2\right)+\frac{7}{8}\left(u_3^2+u_{n-2}^2\right)+\sum_{i=4}^{n-3}u_i^2\right)} \quad (7)$$

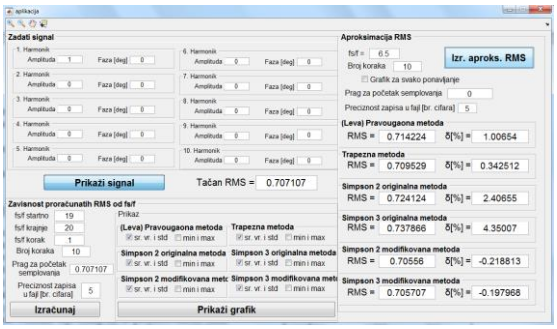
gde je *n* broj odbiraka uzetih za aproksimaciju.

3. OPIS SIMULACIJE

Cilj sotvera je da za zadati signal izračuna tačnu vrednost RMS i ispita kolika je tačnost, tj. greška svake od šest metoda za aproksimaciju RMS predstavljenih u prethodnom delu teksta, i to u zavisnosti od sledećih parametara:

1. Samog zadatog signala
2. Naponskog praga koji definiše početak i kraj periode, tj. semplovanja
3. Odnosa frekvencije semplovanja i frekvencije zadatog signala *fs/f*
1. Broja koraka (proračuni se simuliraju za redom više početnih trenutaka, ekvidistantno uzetih između početka periode zadatog signala i početka periode semplovanja)

Prikaz korisničkog interfejsa projektovanog softvera dat je na Slici 6.



Slika 6. *Prikaz korisničkog interfejsa projektovanog softvera*

Pojedini rezultati simulacije se ispisuju na određenim poljima na interfejsu, pojedini se predstavljaju grafički, a pojedini skladište u izlazne datoteke zarad daljih ispitivanja.

3.1. Zadati signal

Signal na kojem se simulira semplovanje i sprovode analize se zadaje preko amplituda i faza svojih prvih deset harmonika, u gornjem levom delu interfejsa.

Ispod toga se nalazi dugme za sračunavanje njegove tačne vrednosti RMS, kao koren iz polovine sume kvadrata amplituda svakog harmonika. Takođe se pritiskom na to dugme i prikazuje jedna perioda signala u novootvorenom prozoru.

Jasno je da se za različite zadate signale dobijaju različite aproksimacije po svim metodama, različitih tačnosti, odnosno grešaka.

3.2. Naponski prag koji definiše početak i kraj periode

Semplovanje se vrši ekvidinstantno sa frekvencijom *fs* zadatom preko *fs/f* (odnosa frekvencije semplovanja i frekvencije zadatog signala), na jednoj periodi zadatog signala. Perioda signala se "detektuje", kao što je to slučaj i u konkretnoj tehnici koja se u praksi koristi za merenje RMS nepoznatih signala, poređenjem semplovanih vrednosti sa odabranim naponskim pragom koji definiše početak i kraj periode i semplovanja, uz pretpostavku da signal u toku svoje jedne periode prolazi kroz prag samo jedanput u rastućem i jedanput u opadajućem trendu (kao što je to npr. slučaj sa prostoperiodičnim signalom). Prolaz u rastućem trendu, tj. prvi odbirak koji je veći od praga, tako da je njemu prethodni bio manji ili jednak pragu, predstavlja prvi odbirak za proračun, i potom se uzimaju redom svi odbirci dok se ne desi opet isti slučaj, da je jedan odbirak manji ili jednak pragu, a drugi veći od

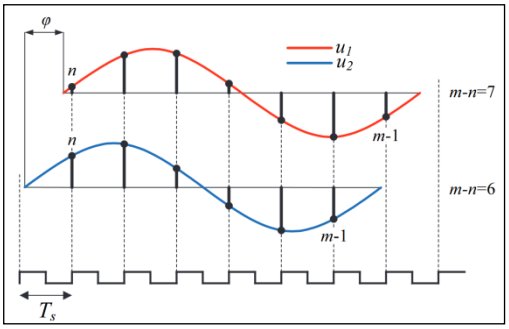
praga. Potom se uzimaju još i taj prvi odbirak veći od praga, koji se koristi za trapeznu i originalne Simpsonove metode, i još sledeća dva odbirka, koji su potrebni za predstavljene modifikovane Simpsonove metode.

Na osnovu opisanog algoritma za "detekciju" periode zadatog signala, lako se može zaključiti da postoje signali za koje ovaj algoritam neće dobro detektovati periodu, što su signali koji više puta prolaze kroz zadati prag, i za takve signale ovaj softver nije predviđen.

Dosadašnje analize [3,1] su pokazale da se po svim metodama za aproksimirani RMS dobija mnogo manja greška ukoliko se za prag uzme vrednost približna pravoj vrednosti RMS zadatog signala, nego recimo 0, i to se potvrđuje simulacijama ovog alata.

3.3. Odnos frekvencije semplovanja i frekvencije zadatog signala *fs/f*

Zadati odnos *fs/f* direktno definiše broj odbiraka na osnovu kojeg se vrši semplovanje zadatog signala. S obzirom da, kao i u praksi, ovaj odnos ne mora biti celobrojan, može se desiti da samo zavisno od trenutka kada se počne sa semplovanjem nekad imamo više a nekad manje odbiraka, kao što je prikazano na Slici 7, što značajno utiče na proračun RMS po bilo kojoj metodi.



Slika 7. *Prikaz kako za isti signal, fs/f i prag, samo zavisno od početnog trenutka semplovanja, nekad možemo imati npr. sedam, a nekad šest odbiraka koji odgovaraju jednoj periodi*

Stoga se, radi što bolje analize prosečnih performansi metoda, proračuni u softveru simuliraju za redom više početnih trenutaka, ekvidistantno uzetih između početka periode zadatog signala i početka periode semplovanja *Ts*, čiji broj se definiše u interfejsu pod poljem "Broj ponavljanja". Proračun RMS se vrši za svaki ovako odabran trenutak, za svaku od šest predstavljenih metoda, i rezultati se u vidu matrice brojnih vrednosti (zadate preciznosti) beleže u fajl pod imenom "Rezultati 1.txt", a u donjem desnom delu intefejsa se prikazuju srednja vrednost proračunate RMS za svaku metodu, kao aritmetička sredina izračunatih RMS za svaki od navedenih početnih trenutaka, a desno od njih se prikazuje i njihova relativna greška (u odnosu na prethodno opisano teoretski izračunatu tačnu vrednost RMS zadatog signala). Ukoliko je broj odbiraka takav da originalne Simpsonove metode ne mogu da se primene, ti slučajevi se ne uzimaju za proračun srednje vrednosti RMS.

Softver ima opciju i za vršenje analize tačnosti metoda za zadati interval vrednosti odnosa *fs/f*, na osnovu koje se može bolje shvatiti i predvideti ponašanje metoda za proizvoljan odnos *fs/f*. Parametri koji definišu početak, kraj i korak ovog intervala se zadaju u donjem levom delu interfejsa. Ispod njih se nalazi dugme "Prikaži", pritiskom na koje se na osnovu zadatog intervala za svaki zadati korak, grafički prikazuju:

1. Proračunate (srednje) vrednosti RMS po svim metodama, kao i tačna vrednost RMS
2. Standardna devijacija za proračunate RMS
3. Minimalna i maksimalna izračunata vrednost za svaku RMS

Uporedo se dobijeni rezultati beleže i u fajl pod nazivom "Rezultati 2.txt".

4. ZAKLJUČAK

Projektovani softver omogućuje brzo i lako ispitivanje performansi navedenih šest metoda za aproksimaciju RMS zadatog signala, u zavisnosti od nekoliko ključnih parametara, poput odnosa frekvencije semplovanja i frekvencije zadatog signala *fs/f*, i naponskog praga koji definiše početak semplovanja, te će se koristiti kao alat za buduća istraživanja na ovu temu.

Za sada imamo neke zaključke za prostoperiodičan oblik signala, a za složenoperiodične slede dalja istraživanja korišćenjem ovog alata.

5. LITERATURA

[1] M. Bulat, S. Mirković, D. Pejić, M. Urekar, Đ. Novaković i N. Gazivoda, "Primena numeričkih metoda integracije na računanje efektivne vrednosti", *ETRAN 2019*, Srebrno jezero, Srbija, 2019.

[2] M. Bulat, S. Mirković, D. Pejić, P. Sovilj, M. Urekar, Đ. Novaković i N. Gazivoda, "Modifikacija Simpsonovih Pravila Numeričke Integracije Pri Određivanju Efektivne Vrednosti Napona", *Kongres metrologa*, Šabac, Srbija, Oktobar 2019,

[3] S. Mirković, D. Pejić, M. Urekar, B. Vujičić, Đ. Novaković, "Improvement of an Existing Method of Asynchronous Sampling for Determining RMS Value", *Serbian Journal Of Electrical Engineering*, Vol. 15, No. 1, Februar 2018

Kratka biografija:

Miloš Cicmil rođen je u Vrbasu 1993. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Instrumentacija i merenje odbranio je 2019.god.

kontakt: miloscicmil@rocketmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.8
DOI: https://doi.org/10.24867/07BE44Mladenovic

PRIMENA UČENJA SA USLOVLJAVANJEM ZA OBUČAVANJE AGENTA ZA AUTONOMNU VOŽNJU AUTOMOBILA U SIMULATORU AIRSIM

USING REINFORCEMENT LEARNING TO TRAIN AN AGENT FOR AUTONOMOUS DRIVING IN THE AIRSIM SIMULATOR

Miloš Mladenović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Učenje uslovljavanjem je napretkom dubokog učenja i hardvera kao i razvojem novih naučno-tehnoloških izazova kao što su razvoj softvera za robote i samovozeće automobile postalo plodno istraživačko tle za sve naučnike i inženjere zainteresovane za ovu oblast. U ovom radu predstavljen je drugačiji pristup problemu autonomne vožnje u simulatoru - u kom se kombinuju tehnike računarskog vida i učenja sa uslovljavanjem da se napravi agent koji će se uspešno kretati u simuliranom okruženju. Evaluacija agenta urađena je poređenjem performansa modela u odnosu na postizanje zadatog cilja.*

Ključne reči: *Učenje uslovljavanjem, autonomna vožnja, simulacija, računarski vid, nagrade, neuronske mreže*

Abstract – *With the advancement of deep learning and hardware, along with emergence of new technological challenges like software for robots and self-driving cars – reinforcement learning has become a fertile ground for researchers interested in this field. In this paper a novel approach has been proposed to solving the problem of autonomous driving in simulator, which combines techniques of computer vision and reinforcement learning to create a software agent that can drive a car successfully in a simulation. Evaluation of the agent has been done by comparing results and expected/given goal.*

Keywords: *Reinforcement learning, autonomous driving, simulation, computer vision, rewards, neural networks*

1. UVOD

U poslednjih nekoliko godina veliki napredak na polju razvoja grafičkih čipova i njihove procesorske moći doveo je do toga da je veliku količinu podataka sada moguće lakše obraditi, pa su duboke neuronske mreže (eng. *Deep Neural Networks*) doživele veliku ekspanziju, a sa njima i duboko učenje (eng. *Deep Learning*) i time zavladale poljem veštačke inteligencije i mašinskog učenja.

Jedna od oblasti veštačke inteligencije kojoj je razvoj dubokog učenja doneo najviše napretka je učenje sa uslovljavanjem (eng. *Reinforcement learning)*, čime je omogućeno da se klasični pristup interakcije agenta i okruženja – kretanje uz dobijanje nagrada i kazni podigne na novi nivo i obradom podataka kroz neuronske mreže dođe do optimizovanijih rešenja i unapređenih algoritama.

Polje na kom poslednjih godina veštačka inteligencija pokazuje svoj puni potencijal i ima najveći uticaj jeste tehnologija samovozećih automobila. Kako bi se prikupila ogromna količina podataka potrebna za obučavanje samovozećih automobila potrebno je posebno opremljenim vozilima preći milione kilometara po svim uslovima vožnje, što je vremenski i po pitanju resursa veoma zahtevno. Često su podaci potrebni brže i u drugačijem obliku od onog koji je trenutno dostupan, pa su napredni računarski simulatori vožnje omogućili olakšavanje ovog zadatka. Jedan od njih je i Microsoft-ov „*AirSim*“ – simulator za testiranje i vožnju automobila i kvadkoptera.

Tema ovog rada je kombinacija istraživanja u oblastima navedenim iznad – upotreba algoritama učenja sa uslovljavanjem za obučavanje modela koji upravlja automobilom u okruženju „*AirSim*“ simulatora.

2. POSTOJEĆA REŠENJA

Postoji više rešenja koja su se poslednjih godina bavila problemom učenja sa uslovljavanjem i agenta koji bi ga koristio da vozi automobil u simulatoru. Jedna od najpopularnijih platformi za simulaciju kretanja i trke automobilima je *Torcs* [1], koja je korišćena za razvijanje brojnih autonomnih agenata. Neki od primera algoritama razvijenih u ovom simulatoru su *Monte Carlo tree search* [2], evolucioni algoritmi [3] i *Q-learning* [4]. Sem *Torcs-a*, projekat CARMA je bio inspirisan napretkom *DeepMind*-a sa DQN algoritmom u *Atari* okruženju i rešili su da skaliraju algoritam na okruženje *Vdrift* [5] gde su diskretizovali prostor akcija da bi se prilagodio DQN algoritmu. Korišćenjem ručno napravljene proste funkcije nagrade zajedno sa podacima sa senzora i slika uspeli su da dobiju rezultate koji su nadmašili ručno napravljen i programiran kontroler okruženja u tri kategorije: prosečna nagrada, prosečna brzina i maksimalna brzina.

Jedan *Microsoft*-ov istraživački tim se, u sklopu razvoja simulatora *AirSim* i promocije njegovog ekosistema i mogućnosti u kombinaciji sa treniranjem na *Azure cloud* infrastrukturi, bavio i problemom učenja sa uslovljavanjem i njegove primene na kreiranje agenta za autonomnu vožnju u simulacionom okruženju *Neighborhood*. [6] Oni su razvili model treniran distribuiranim dubokim učenjem sa uslovljavanjem, koji je koristio moć računara u oblaku. Model je zasnovan na *DQN* algoritmu *DeepMind-a*, ali je i koristio prednosti *transfera učenja*, odnosno težine u konvolutivnim slojevima neuronske mreže korišćene unutar *DQN-a* su već bile modifikovane i

---

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

unapređene nadgledanim učenjem, odnosno prethodnom vožnjom agenta po okruženju u *AirSim* simulatoru.

3. METODOLOGIJA

Ova sekcija je zadužena za opisivanje okruženja sa kojim agent vrši interakciju, alata i gotovih algoritama korišćenih u implementaciji. Takođe, biće dat i pristup problemu sa aspekta računarskog vida i napomena o korišćenim istog. Agent je implementiran u programskom jeziku *Python* [3].

3.1. Microsoft *AirSim* simulator

U ovom radu, kako bi se obezbedilo okruženje za treniranje korišćen je *Microsoft AirSim – open-source* simulator za automobile i dronove, baziran na *Unreal Engine*-u. Razvijen je kao više-platformski simulator otvorenog koda, koji podržava i hardversku kontrolu pomoću kontrolera kao što je PX4, za fizički i vizuelno realistične simulacije. Poseduje nekoliko API-ja za dobijanje podataka o stanju simulacije, kao i kontrolu parametara okruženja i načina kretanja i parametara vozila koje se kontroliše. [7] Na *slici 1* prikazano je *Hawaii* okruženje u *AirSim*-u.

Slika 1: *Okruženje Hawaii u AirSim simulatoru*

3.2. *OpenAI baselines* i *Gym*

Neprofitna organizacija *OpenAI* ponudila je kroz otvoreni kod skup visoko kvalitetnih implementacija algoritama za učenje sa uslovljavanjem, koji mogu da služe istraživačima u ovoj oblasti da lakše definišu i identifikuju nove ideje i koji služe kao dobra osnova za dalji razvoj i unapređenje oblasti učenja sa uslovljavanjem. Ovaj projekat, nazvan *baselines*, tj. *osnove*. Kako bi se novo okruženje, kao što je npr. korišćeni i ranije pomenuti *Hawaii* iz *AirSim-a* prilagodilo upotrebi kod gore-navedenih algoritama iz paketa *baselines*, potrebno je oblikovati ga na odgovarajući način, a to se postiže pomoću *gym* skupa alata i pravila. Glavni interfejs *Gym* seta je *Env*, koji predstavlja ujedinjeni interfejs okruženja, a njegove glavne metode, koje treba implementirati pri kreiranju sopstvenog modela okruženja su: *reset()* – resetuje okruženje na početno stanje; *step(action)* – izvršava zadatu akciju u okruženju I vraća sliku stanja, nagradu I sledeće stanje; *render(mode="human")* – iscrtava okruženje, tj. prikazuje sliku okruženja u prozoru. Ključni atribut koji treba zadati je *action_space* – određuje prostor akcija koje agent može da izvrši.

3.3. Pristup problemu iz ugla računarskog vida – detekcija središnje linije puta

Dobra nagrada je glavni pokretač i osnov uspešnosti agenta za učenje sa uslovljavanjem, pa je stoga njena optimizacija ključ dobrih rezultata treniranja. U nekim radovima, kao što je [8], osnov za računanje nagrade, kod automobila koji se autonomno kreću po nekoj stazi u simuliranom okruženju a koriste učenje sa uslovljavanjem kao algoritam koji ih „pogoni", bila je udaljenost vozila od linija na putu, odnosno od središnje linije puta ili krajnje desne, odnosno leve linije koje ograničavaju traku. Ovaj pristup se pokazao kao korektan, međutim veliki problem da se algoritam sa takvom vrstom nagrade upotrebi bilo gde drugde, na drugoj stazi, okruženju ili čak i u realnim uslovima, jeste to što okruženje i staza/put moraju biti mapirani i da se poznaju koordinate svih tačaka na putu, ali i širina puta, ili pak lokacija središnje linije puta, zavisno od pristupa.

Rešenje opisanog problema je upotreba tehnika *računarskog vida* kojima se vrši detekcija središnje linije na putu na slici pribavljenoj sa kamere postavljene na prednjoj strani vozila i onda optimizovala nagrada da se prati detektovana linija. U narednih nekoliko rečenica, biće opisani koraci korišćeni da se dođe do detektovane linije:

1. Primeniti *Gausov blur* filter na sliku pribavljenu sa kamere.
2. Promena modela boja rezultujuće slike iz prethodnog koraka – prebacivanje iz RGB u HSV spektar, kako bi se lakše definisao raspon žute boje, koja čini središnju liniju u traci na putu.
3. „Maskiranje slike" – uklanjanje svih boja koje ne spadaju u definisani raspon datih vrednosti za HSV sprektar.
4. Korišćenje *Canny edge detetection* [9] za detekciju ivica na maskiranoj slici. Primer rezultata nakon ove transformacije dat je na *slici 2*

Slika 2: *Slika puta i okoline nakon maskiranja HSV spektra za žutu i detekcije ivica*

5. Nakon detekcije ivica, konačno se primenjuje klasična *Hafova* transformacija za detekciju linija na slici, kojom se dobija niz x, y koordinata linija detektovanih na slici, odnosno njihove početne i krajnje tačke. Na s*lici 3* dat je rezultat nakon ovog koraka u stazi *Hawaii*, korišćenoj u ovom radu.

Slika 3: *Središnja linija na putu detektovana Hafovom transformacijom*

3.4. Implementacija rešenja

Implementaciju *Gym* okruženja je moguće izvesti tako da podržava i kontinualne i diskretne akcije. Za različite isprobane algoritme, korišćen je drugačiji tip akcija, odnosno kod DQN algoritma je korišćen diskretan skup akcija, a kod *policy gradient* kontinualan. Diskretan skup akcija podrazumeva određen krajnji broj vrednosti za

uglove skretanja u rasponu [-1, 1]. Kod DQN algoritma sa najboljim dosadašnjim rezultatima npr. korišćen je sledeći skup akcija: [-0.65, -0.5, -0.25, -0.1, 0.0, 0.1, 0.25, 0.5, 0.65]. Zbog unapred poznatih detalja o okruženju - stazi *Hawaii*, odnosno da staza ne sadrži raskrsnice i neke isuviše oštre krivine, upotrebljen je skup akcija koji ne sadrži neke ekstremne vrednosti za uglove skretanja kao što su -1 i 1 - kako bi kretanje automobila po stazi bilo uglađeno i bez previše naglih pokreta.

Pored navedenog skupa akcija, po *Gym* specifikaciji, potrebno je implementirati i prostor stanja okruženja. Pod ovim se podrazumeva oblik strukture podataka koja će biti korišćena za predstavljanje okruženja, tip podataka, kao i najveća i najmanja vrednost koja se može javiti.

Implementirano okruženje za prostor stanja koristi preprocesiranu sliku koju dobija iz okruženja, sa kamere koja se nalazi na prednjem braniku automobila i usmerena je ka putu, tako da snima trake u kojima se vozilo kreće i malo stvari u okolini. Pod preprocesiranjem se podrazumeva sečenje slike po širini i dužini na neke eksperimentalno utvrđene idealne dimenzije .

3.5. Računanje nagrade agenta

Računanje nagrade je ključna stvar koja određuje uspešnost treniranja agenta za učenje sa uslovljavanjem. Ono se izvršava u svakom vremenskom koraku, što je kod implementiranog agenta svaki frejm.

Nagrada je kod implementiranog agenta spoj nekoliko komponenti, odnosno sadrži ciljeve koje prilikom vožnje mora da ispuni - da vozilo stigne od tačke A do tačke B, usput se držeći središnje linije puta, prateći put željenom brzinom i izbegavajući sve prepreke na tom putu. Zbir ovih komponenti sačinjava nagradu agenta, a najvažnija je ona koja ima ključnu ulogu u kretanju – praćenje središnje linije puta.

Kako bi se držanje središnje linije puta uračunalo u nagradu, potrebno je odrediti trenutnu udaljenost automobila od nje, a to je učinjeno tako što su detektovane sve linije algoritmom opisanim u *3.3*, nakon čega je određivano najbliže rastojanje između detektovanih linija i trenutne pozicije vozila (koja je uvek na sredini slike). Prilikom detekcije linije, eksperimentalno je utvrđeno da je zbog senki i položaja sunca u nekom trenutku moguće menjanje boja na slici tako da nije moguće inicijalno detektovati središnju liniju jer je zašla u previše tamni spektar žute. Kada bi se ovo desilo, da treniranje ne bi bilo prekinuto odmah i bilo računato da je auto krenuo u „*off-road*“ režim kretanja, odnosno van puta, povećavala bi se osvetljenost slike za određeni stepen, da bi se na taj način probala detekcija linije ponovo. Ukoliko i nakon toga nije bilo detektovane središnje linije, znači da je udaljenost od nje maksimalna i da je automobil krenuo u kretanje van puta, što mu je dozvoljeno određen broj vremenskih koraka.

Nakon što je određeno koja je najbliža detektovana linija trenutnoj liniji kretanja vozila, to rastojanje se podeli najvećim mogućim rastojanjem između dve linije (nešto manje od širine slike) i odradi eksponencijalna funkcija, kako bi se nagrada skalirala u rasponu od 0 do 1.

U formuli *1* dat je način računanja nagrade. Primenom ove tehnike računarskog vida, agent može da se kreće i trenira u okruženju bez bilo kakve ljudske pomoći.

$$R = \begin{cases} e^{-d_{min}} + R_v + R_{cg}, & za\ d_{min} > -1 \\ x * n, za\ d_{min} = -1\ i\ n < n_{max} \end{cases} \quad (1)$$

Parametri u gore navedenoj formuli označavaju sledeće:

- Parametar *d* je udaljenost od najbliže detektovane linije na putu
- Parametar *n* je broj frejmova (koraka) u kojima je automobil proveo van puta
- Parametar n_{max} je maksimalni broj frejmova dozvoljen agentu da se kreće van puta (tj. kada središnja linija nije detektovana) i zadat je programski
- Parametar x predstavlja nagradu (kaznu) za kretanje agenta van puta
- R_{cg} je parametar koji karakteriše uticaj približavanja cilju, odnosno ukoliko je vozilo u trenutnom koraku izvršavanja bliže cilju nego što je bilo u prethodnom koraku, dobiće nagradu. Ovo je zadato formulom *2*

$$R_{cg} = \begin{cases} R_g, & za\ dist_{curr} < dist_{prev} \\ 0, & za\ dist_{curr} \geq dist_{prev} \end{cases} \quad (2)$$

- Parametar R_s je nagrada, odnosno kazna za kretanje brzinom većom od maksimalne ili manjom od minimalne

Postoje dva specijalna slučaja nagrade, koji omogućavaju završetak epizode treniranja pre dostizanja zadatog broja koraka: R_g kada se dostigne cilj zadat GPS koordinatama i kada se automobil sudari sa nekom preprekom – R_c.

3.6. Primenjeni algoritmi

Algoritmi koji su korišćeni za treniranje agenta učenja sa uslovljavanjem dolaze iz paketa *baselines*[1] i isprobani su algoritmi DQN i unapređene verzije *Policy gradient* algoritma. *DQN* sa svojim poboljšanjima, kao i *Policy gradient*. S obzirom na to da je DQN agent ranije opisivan i u ovom slučaju pokazao bolje rezultate u *tabeli* 1 će biti predstavljeni parametri treniranja za DQN algoritam, zajedno sa kratkim opisom i vrednošću korišćenom u verziji algoritma koji je pokazao najbolje rezultate.

Tabela 1: *Parametri DQN algoritma*

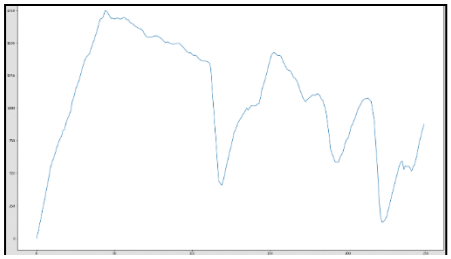
Parametar	Značenje	Korišćena vrednost
network	neuronska mreža koja se koristi kao aproksimator funkcija. Može biti *cnn, conv_only* ili *mpl*	*cnn*
lr	stopa učenja za Adam optimizator	0.0011
buf_size	veličina bafera za reprodukciju	75000
exp_fra	*exploration-exploitation rate*	0.1
exploration_final_eps	konačna vrednost verovatnoće nasumičnog odabira akcija	0.015

4. EKSPERIMENTALNI REZULTATI I DISKUSIJA

Zbog kompleksnosti problema i mnogo parametara prisutnih na slici, treniranje agenta za upravljanje auto-

[1] https://github.com/openai/baselines.

mobilom u *AirSim*-u sa primenjenim tehnikama i implementacijama opisanim u prethodnim poglavljima, trajalo je i po dvadesetak sati, da bi se dobio model koji stiže do cilja i pokazuje solidnu voznu dinamiku pritom, držeći se glavne vodilje – držanje srednje linije puta. Na *slici* 4 prikazana je oscilacija, rast i ponašanje nagrade za vreme treniranja, na primeru od 300 000 vremenskih koraka.



Slika 4: *Nagrada po broju vremenskih koraka*

Kao što možemo da vidimo sa slike, nagrada stabilno raste, kako prolazi vreme, dok ne dođe do neke maksimalne vrednosti od ~1750, kada otprilike agent stiže blizu cilja kretanja, pa bi naredni koraci pri treniranju i poboljšavanju nagrade trebalo da budu da se kreće što brže i što više drži središnje linije. Međutim, vidimo da kasnije nagrada opada, čemu je najverovatniji uzrok promena parametara okoline, kao što su pad nivoa osvetljenosti od sunca i promena vremena u danu, koji je u simulatoru nešto sporiji nego u realnom vremenu, ali opet jedan od glavnih faktora realističnosti *AirSim-a*. Pošto dolazi do promene osvetljenja i boja u okolini i na putu, agent više ne može da detektuje središnju liniju i počinje loše da se kreće i sudara sa okolinom, tako da nagrada drastično pada. Uprkos tome što nagrada vremenom počne da opada, najbolji model se kroz *callback* ipak čuva i moguće je proveriti njegove performanse. Snimak procesa treniranja, kao i ponašanja i vožnje nekih od treniranih modela moguće je naći na sledećem linku:
https://drive.google.com/open?id=1pFkB9v0DRPdPfEfH6lZwDcxU2DPV1gKg
Osmišljavanje nagrade izložene u ovom radu je dobra početna tačka, ali poboljšanja bi moglo da bude i sa aspekta kretanja automobila u traci, da se detektuju i linije na stazi koje ograničavaju traku sa leve i desne strane i da se vozilo, umesto duž središnje linije kreće u svojoj traci pravilno. Ovo je i bila inicijalna namera, međutim zbog nedostatka dobrih staza i besplatnih okruženja za *AirSim* simulator, *Hawaii* je ostala jedina sa adekvatnim putnim oznakama, koje zadovoljavaju inicijalnu ideju.

5. ZAKLJUČAK

Oblast mašinskog učenja sa uslovljavanjem, zahvaljujući sve boljem hardveru na kom se mogu obučavati algoritmi koji obrađuju veliku količinu podataka visokih dimenzija i napretku u simulatorima koji sve više elemenata realnog sveta prenose u virtuelni, ima veliku mogućnost napredovanja i uticaja na svet oko nas. U ovom radu dat je kratak pregled oblasti i njenih najvažnijih modernih algoritama, koji su na kraju i primenjeni na rastuću i sve značajniju oblast autonomne vožnje.

Što se tiče samog agenta razvijenog i predstavljenog u ovom radu, postoji mnogo prostora za poboljšanja i unapređenja, pre svega u vidu smanjenja uticaja svetlosti i promene doba dana na detektovanu liniju, pa samim tim i na nagradu i performanse modela. Druga stvar koju bi trebalo poboljšati je i neujednačeno kretanje agenta koji često „oscilira“ levo-desno oko centralne linije, na šta verovatno utiče diskretizovani skup akcija, koji bi moglo povećati ili staviti da bude kontinualan, ali i primeniti druge algoritme kao što je *DDPG* i sl.
Bez obzira na to što počeci učenja sa uslovljavanjem dosežu do ranih 60-ih godina prošlog veka, skorašnji napredak na polju dubokog mašinskog učenja i razvitak hardvera omogućili su procvat ove oblasti, koja pretenduje da u skorijoj budućnosti utaba put ka generalnoj veštačkoj inteligenciji, više nego oblasti nadgledanog i nenadgledanog učenja, ukoliko se dovoljno istraživačkog napora uloži u to.

6. LITERATURA

[1] Loaiacono i Cardamone, „Simulated car racing championship: Competition software manual,“ 2013.

[2] F. J, F. N, Vielwerth i T. J, „ Monte-Carlo Tree Search for Simulated Car Racing,“ 2015.

[3] Koutnik, Cuccu i Schmidhuber, „Evolving large-scale neural networks for vision-based reinforcement learning“.

[4] Loiacono, Prete, L. P. i C. L, „Learning to overtake in torcs using simple reinforcement leraning“.

[5] M. V. A. N, „Carma: A deep reinforcement learning approach to autonomous driving“.

[6] M. Spryn, S. Aditya i P. Dhawal. [Na mreži]. Available: https://github.com/microsoft/AutonomousDrivingCookbook/tree/master/DistributedRL.

[7] Microsoft. [Na mreži]. Available: https://microsoft.github.io/AirSim/docs/apis/.

[8] K. M i K. S, „Autonomous vehicle control via deep reinforcement learning,“ Master's thesis, 2017.

[9] R. Szelinski, Computer Vision: Algorithms and Applications, Springer, 2011.

Kratka biografija:

Miloš Mladenović je rođen 24.05.1994. u Gnjilanu, Republika Srbija. Osnovnu školu „Desanka Maksimović“ završio je 2009. godine u Kosovskoj Kamenici. Gimnaziju u istom gradu završava 2013. godine i upisuje osnovne akademske studije Elektronskom fakultetu u Nišu. Zvanje diplomirani inženjer elektrotehnike i računarstva stiče 2017. godine, sa prosečnom ocenom 9.71, uz specijalizaciju računarske nauke i informatika. Nakon toga, iste godine, upisuje master akademske studije na Fakultetu tehničkih nauka u Novom Sadu, odsek Elektrotehnika i računarstvo, smer Računarstvo i automatika, modul Inteligentni sistemi. Položio je sve ispite predviđene planom i programom master studija uz prosečnu ocenu 9.71.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/07BE45Savic

REALIZACIJA PAMETNIH MREŽA I DISTRIBUIRANI GENERATORI

SMART GRID REALISATION AND DISTRIBUTED GENERATORS

Isidora Savić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu opisana je ideja pametne mreže, sa svim svojim prednostima i nedostacima. Izvršeno je upoređivanje postojećeg elektroenergetskog sistema sa „pametnim". Posebna pažnja posvećena je zastupljenosti obnovljivih izvora energije u svetu i u Srbiji, kao i tendencija ka njihovom maksimalnom iskorišćenju. Predstavljen je pametni distributivni sistem sa ulogama distribuiranih generatora u njemu. U radu je takođe predstavljena zaštita u distributivnim mrežama, kao i analiza uticaja distribuiranih generatora na rad relejne zaštite, nivo struje kratkog spoja i na pouzdanost napajana. Dobijeni rezultati su predstavljeni tabelarno i kroz dijagrame.*

Ključne reči: *Pametna mreža, Distributivna mreža, Distribuirani generator, Relejna zaštita*

Abstract – *This paper describes the idea of a smart grid, with all its advantages and disadvantages. The comparison of the existing electricity system with the "smart" one was made. Special attention is paid to the presence of renewable energy sources in the world and in Serbia, as well as the tendency towards their maximum utilization. A smart distribution system was introduced with the roles of distributed generators in it. The paper also presents protection in distribution networks, as well as an analysis of the influence of distributed generators on the operation of relay protection, the level of short circuit current and the reliability of the power supply. The obtained results are presented in tables and diagrams.*

Keywords: *Smart grid, Distribution network, Distributed generator, Relay protection*

1. UVOD

Potrebe potrošača za električnom energijom su široke i raznovrsne, jer je električna energija najplemenitiji vid energije u prirodi. U elektranama se vrši pretvaranje jedne vrste energije (toplotne energije, kinetičke energije vode, energije vetra itd.) u električnu energiju. U zavisnosti od načina pretvaranja energije, razlikuju se termoelektrane, hidroelektrane, vetroelektrane itd.

Elektroenergetski sistem (EES) je sistem čiji je osnovni zadatak da osigura kvalitetnu isporuku električne energije uz minimalne troškove. Ovakav postojeći EES suočava se sa velikim ekološkim, finansijskim i tehničkim izazovima. Upravo zbog toga, javlja se potreba za njegovom modernizacijom. Javlja se ideja za izgradnjom pametne električne mreže (Smart Grid) u cilju značajnog poboljšanja pouzdanosti, energetske efikasnosti, sigurnosti i kvaliteta napajanja električnom energijom.

Prilično je izvesno da će pametne mreže biti, a i u određenoj meri već jesu, deo svakodnevnice većine stanovnika u urbanim sredinama.

U drugom delu rada je data struktura tradicionalnog elektroenergetskog Sistema (EES), a u trećem obnovljivi izvori energije (OIE). Četvrti deo je posvećen ideji pametnih mreža, a peti trendu razvoja pametnih mreža u Evropi. U šestom delu su navedene dobiti od primene distribuiranih generatora, a poslednja dva dela su zaključak i literatura za pisanje ovog rada.

2. INFRASTRUKTURA TRADICIONALNOG EES

EES su uglavnom građeni pre 50 godina. Princip prema kojem su građeni je bio taj da se električna energija proizvodila u velikim elektranama, odakle je preko transformatora prenošena u visokonaponsku elektroenergetsku mrežu. Prenosna mreža je korišćena za transport električne energije do distributivnih transformatora, često na velike udaljenosti. Od distributivnih transformatora je električna energija prenošena preko srednjenaponskih i niskonaponskih električnih mreža do krajnjih potrošača.

2.1. Problemi postojećeg EES i elektroprivrede

Organizacija elektroprivrede, grane privrede koja se bavi eksploatacijom elektroenergetskog sistema, zavisi od tržišta električne energije i od vlasništva nad elektroenergetskim sistemom.

Koncept organizacije elektroenergetskog sektora podrazumeva da su proizvodnja, prenos i distribucija (sa prodajom) električne energije organizovane u okviru jednog preduzeća, najčešće državnog. Razlog primene takvog koncepta je verovanje da je situacija na tržištu električne energije takva da je samo jedno preduzeće u stanju da profitabilno nudi usluge (u slučaju konkurencije, samo bi jedno preduzeće bilo u stanju da dugoročno opstane). Elektroprivrede uglavnom nisu u stanju da same finansiraju razvijanje novih, potrebnih tehnologija i uređaja, kao i velike infrastrukturne promene u mreži.

To znači da je budućnost koncepta pametnih mreža veoma zavisna od spremnosti same država da ulažu u njen razvoj i primenu. Ovakav pristup ima brojne nedostatke, a najveći je zanemarivanje planiranja i razvoja, upravljanja finansijama i troškovima itd.

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

Rešenje postojećih problema potrebno je tražiti u odgovornom i održivom razvoju, jer uobičajeni način rada više nije opcija. Vlade razvijenih zemalja donose brojne uredbe i inicijative o energetskoj efikasnosti, konzervaciji energije, smanjenju zagađenja, smanjenju uticaja elektroenergetskog sistema na klimatske promene itd. Jedan od imperativa desetogodišnje strategije ekonomskog oporavka i razvoja zemalja Evropske unije „Evropa 2020" (eng. *Europe 2020*) je ostvarenje takozvanih ciljeva „20-20-20" [1]:

• do 2020. godine smanjiti emisiju gasova koji izazivaju efekat staklene bašte za 20% u odnosu na nivo iz 1990. godine,
• energija iz obnovljivih izvora učestvuje sa 20% u strukturi energije Evropske Unije,
• ukupna primarna potrošnja energije u EU se mora smanjiti za 20%.

3. ZASTUPLJENOST OBNOVLJIVIH IZVORA

Usled socijalnog i ekonomskog uticaja, energetske kompanije su pod velikim pritiskom da budu što efikasnije, kako bi postigle ne samo konkurentnost sa cenama, nego i adekvatan nivo zaštite životne sredine. Dodatno, kompanije su pod pritiskom da pokažu odgovornost, fleksibilnost, inovativnost i održivost postojećih resursa. Da bi poboljšale profitabilnost, stekle prednost u odnosu na konkurenciju i održale adekvatan nivo zaštite životne sredine, kompanije moraju da naprave izmene u svojoj strategiji poslovanja, sa akcentom na sledeće oblasti [2]: zahtevi za energijom, zastupljenost obnovljivih izvora energije u proizvodnji, smanjenje emisije ugljen-dioksida, razlika u ceni pogonskih goriva i infrastruktura sa različitim tipovima proizvodnje električne energije, poboljšanje proizvodnje nuklearne energije ili prestanak proizvodnje, pritisci za smanjenje cene električne energije, postojeći sistemi i potreba za prilagođavanjem novim uslovima.

3.1. Vrste obnovljivih izvora energije

U nastavku se daje pregled OIE:

• Vodne snage, (energija vodotokova, morskih struja i talasa, plime i oseke).
• Biomasa (i biogas, uključujući i drvo i otpatke).
• Energija Sunčevog zračenja.
• Energija vetra.
• Unutrašnja toplota Zemlje (geotermalna energija).
• Energija plime i oseke.
• Energija talasa.

3.2. Razvoj OIE u Srbiji

Snažan rast se nastavio kako u proizvodnji, tako i u prodaji, ali i u instalaciji postrojenja OIE zahvaljujući promeni politika mnogih zemalja, pa tako i u Srbiji.

Strategijom razvoja energetike Republike Srbije do 2015. godine u okviru Prioriteta selektivnog korišćenja obnovljivih izvora energije posebno je istaknuto da u Republici Srbiji postoje posebne pogodnosti i potrebe za organizovano korišćenje OIE u tzv. decentralizovanoj proizvodnji toplotne (sagorevanjem biomase i „sakupljanjem" Sunčevog zračenja) i električne energije (izgradnjom malih, mini i mikro hidroelektrana i vetrogeneratora snage do 10 MW), za zadovoljenje potreba lokalnih potrošača, kao i za isporuke viškova električne energije lokalnoj mreži u okviru elektroenergetskog sistema Srbije.

Osnovni cilj u oblasti OIE jeste stvaranje uslova za veće korišćenje OIE, odnosno stvaranje uslova za investiranje u OIE. Broj izgrađenih objekata za eksploataciju OIE u Republici Srbiji i njihova aktuelna godišnja energetska produkcija su zanemarljivi. Kapital koji je uložen u do sada izgrađene objekte je male vrednosti. Veoma su mali, gledano sa nacionalnog nivoa, i finansijski rezultati ostvareni radom do sada izgrađenih objekata za korišćenje. Tehničko-tehnološke karakteristike opreme za eksploataciju OIE lošije su od karakteristika slične opreme koja se danas koristi u EU [3].

4. IDEJA PAMETNE MREŽE

4.1. Definicija Smart Grid

Prvu zvaničnu definiciju Smart Grid-a obezbedio je Zakon o energetskoj nezavisnosti i bezbednosti iz 2007. godine, koji je odobrio Kongres SAD u januaru 2007. godine. Naslov KSIII ovog zakona daje opis sa deset karakteristika koje se mogu smatrati definicijom Smart Grid-a i to:

„Politika Sjedinjenih Država jeste da podrži modernizaciju sistema prenosa i distribucije električne energije radi održavanja pouzdane i sigurne infrastrukture za električnu energiju koja može da odgovori na budući rast potražnje i da postigne svaku od sledećih karakteristika koje zajedno karakterišu Smart Grid:

1. Povećana upotreba digitalnih informacija i kontrola tehnologije za poboljšanje pouzdanosti, sigurnosti i efikasnosti električne mreže.

2. Dinamička optimizacija operacija mreže i resursa, uz potpunu ciber-sigurnost.

3. Razvoj i integraciju distribuiranih resursa i generatora, uključujući obnovljive resurse.

4. Razvoj i inkorporiranje odgovora na tražnju, resurse potražnje i resurse energetske efikasnosti.

5. Razmeštanje „pametnih" tehnologija (u realnom vremenu, automatizovane, interaktivne tehnologije koje optimizuju fizičko funkcionisanje aparata i potrošačkih uređaja) za merenje, komunikacije u vezi sa mrežnim operacijama i statusom i automatizaciju distribucije.

6. Integracija „pametnih" uređaja i potrošačkih uređaja.

7. Razmeštanje i integracija naprednih tehnologija za skladištenje električne energije i vrhova za brijanje, uključujući električna i hibridna električna vozila sa električnim i hibridnim električnim grejanjem i klima uređaji za termičko skladištenje,

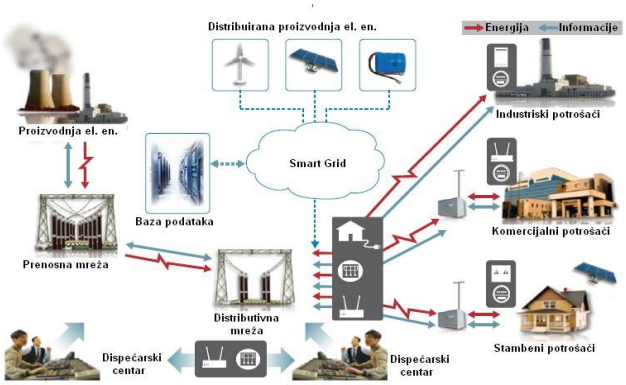
8. Obezbeđivanje pravovremenih informacija i kontrola potrošača.

9. Razvijanje standarda za komunikaciju i interoperabilnost uređaja i opreme povezanih sa električnom mrežom, uključujući i infrastrukturu koja služi mreži.

10. Identifikacija i smanjivanje nerazumnih ili nepotrebnih prepreka za usvajanje inteligentnih mrežnih tehnologija, praksi i usluga."

4.2 Tehnologije pametne mreže

Pametna mreža predstavlja skup različitih tehnologija koje će se koristiti za poboljšanje kvaliteta rada elektroenergetskog sistema i koja će omogućiti: bolju kontrolu postojeće elektroenergetske infrastrukture, dodatnu funkcionalnost i poboljšanje postojeće infrastrukture, integraciju novih postrojenja u postojeći sistem, poboljšanje celokupnog elektroenergetskog sistema.



Slika 4.1 – Koncept pametnih mreža

4.3 Prednosti pametne mreže

Koncept pametnih mreža, pažljivo projektovan i primenjen, u odnosu na tradicionalni pristup, pruža mnogobrojne pogodnosti.

Prednosti pametne mreže većina zagovornika nove tehnologije najčešće prezentuju putem ekonomskih pogodnosti, što je razumljivo, s obzirom na to da su glavni kupci i korisnici nove tehnologije deregulisana elektroprivredna preduzeća koja se „takmiče" na konkurentnom tržištu i kojima je profit najbitnija stavka poslovanja.

Sa ekonomskog aspekta, ostvaruju se sledeće pogodnosti [4]:

- povećana korisnost infrastrukture AU (Asset Utilization),
- kapitalne uštede u prenosu i distribuciji električne energije,
- uštede pri održavanju i eksploataciji objekata prenosa i distribucije električne energije,
- smanjena mogućnost krađe električne energije,
- veća energetska efikasnost,
- manja cena električne energije.

5. TREND RAZVOJA PAMETNIH MREŽA U EVROPI

Iako je postignut veliki napredak u korišćenju obnovljive energije i energetske efikasnosti i širenja pristupa energiji u poslednjoj deceniji, svet nije na putu da ispuni međunarodne klimatske ciljeve, niti međunarodne ciljeve za održivi razvoj.

Utvrđeno je da je potrebna otprilike decenija da bi se globalno zagrevanje održalo ispod ovog nivoa i ida bi se izbegli najgori efekti klimatskih promena. Većina zemalja i dalje subvencionira potrošnju fosilnih goriva, a subvencije za potrošnju fosilnih goriva povećane su za 11% u 2017. godini.

Odmah su potrebne hitne akcije kako bi se promenili naši EES. Šta više, teško je ispuniti ciljeve Ujedinjenih Nacija koji se odnose na održi razvoja 7 (UN Sustainable Development Goal 7) za povećanje obnovljive energije, energetske efikasnosti i pristupa energiji ako se nastavi sadašnji energetski put.

5.1. Pametni distributivni sistemi

Distributivni sistem budućnosti kao deo koncepta pametnih mreža, često se naziva pametni distributivni sistem.

Klasične distributivne mreže projektovane su za jednosmerne tokove snaga (energije) i dimenzionisane samo za potrebe potrošačkih opterećenja. Obnovljivi izvori energije imaju znatno manju energetsku vrednost u poređenju sa fosilnim gorivima zbog čega su njihove elektrane manje veličine. Takve male elektrane priključuju se na distributivnu mrežu i poznate su pod imenom distribuirani izvori električne energije [5].

Podrazumeva se implementacija savremenih informacijskih struktura u električne mreže - daleko većeg obima i složenosti nego što je to slučaj danas. Težnja je da se ovim putem poznaje u svakom trenutku (u realnom vremenu) stanje svakog čvora mreže, snaga koju potrošač zahteva ili je u mogućnosti da da mreži, kao i snaga koju distribuirani izvori u datom trenutku, na osnovu raspoloživih resursa proizvode.

To je jedan složen, dinamički sistem, pa takva mora biti i informacijska mreža koja omogućava razmenu informacija u okviru tog sistema jer predstavlja osnovu za funkcionisanje svih ostalih podsistema. Bitni zahtevi za informacijsku mrežu u ovom slučaju su mogućnost dvosmerne komunikacije između elemenata mreže, putem sigurnih i bezbednih telekomunikacijskih kanala sa mogućnošću velikog protoka informacija.

5.2. Zaštita u distributivnim mrežama

U distributivnim mrežama, prisutan je veliki broj pomoćnih sistema, kako bi se obezbedili zahtevi za pouzdan, siguran i ekonomičan EES. Sistem zaštite ima veoma bitnu ulogu u obezbeđivanju ovih zahteva. Njegova osnovna uloga je da prekine deo mreže sa kvarom i da ograniče mogućnost oštećenja opreme usled kvara. Automatska detekcija i delovanje sistema zaštite je potrebna kako bi se u što kraćem vremenskom roku izolovao kvar i minimizovalo oštećenje opreme [6].

Primenom relejne zaštite omogućava se najbrže moguće isključenje elementa, odnosno dela EES sa kvarom, uz očuvanje funkcionalnosti i održanje stabilnosti ostalog dela sistema. Smanjenje broja i trajanje prekida mogu poboljšati pouzdanost napajanja. Kvalitet napajanja se takođe može poboljšati prilikom brže detekcije kvarova, čime bi se smanjila mogućnost naponskih ulegnuća, treperenja itd.
Relejna zaštita može biti klasifikovana u skladu sa načinom na koji obavlja svoju funkciju.

Postoje: prekostrujni, distantni, diferencijalni, podnaponski (nadnaponski) i drugi releji.

6. ENERGETSKE DOBITI OD DISTRIBUIRANIH GENERATORA

U nastavku se razmatraju benefiti od primene distribuiranih generatora.

6.1. Uticaj distribuiranih generatora na nivo straja kratkog spoja

Uticaj koji će DG imati na struje kratkog spoja u velikoj meri zavisi od sposobnosti DG-a da napaja mesto kvara. Tip generatora koji ima značajan uticaj na nivo struja kratkog spoja je sinhroni generator.

Nije teško ustanoviti da sa porastom snage distribuiranog generatora, sve više raste struja na mestu kratkog spoja, kao i struje po granama distributivne mreže. Ovaj porast vrednosti struja kratkog spoja može dovesti do raznih problema, pre svega u funkcionisanja relejne zaštite, gde može doći do gubitka kako osetljivosti, kao i selektivnosti relejne zaštite.

6.2. Uticaj distribuiranih generatora na rad relejne zaštite

Integracija distribuiranih generatora u sistem bez sumnje menja nivo struja kratkog spoja. Sa druge strane, budući da će se doprinos distribuiranih generatora struji kvara odraziti na struje koje mere uređaji relejne zaštite, postojeći sistem zaštite imaće problema u radu, što će dovesti do toga da će neki kvarovi ostati nedetektovani, a u nekim slučajevima doći će i do problema selektivnosti.

Zbog uticaja distribuiranih generatora na amplitudu i smer struja kratkog spoja, javljaju se sledeći problemi u radu relejne zaštite: 1. problem selektivnosti, 2. skraćenje dosega releja, 3. loša koordinacija rada između reklozera i osigurača, 4. ostrvski rad.

6.3. Uticaj distribuiranih generatora na pouzdanost napajanja

Jedna od koristi upotrebe distribuiranih generatora je baš povećanje pouzdanosti sistema. Upotreba alternativnih izvora napajanja u trenucima kada dođe do kvara ili ispada nekog dela elektroenergetskog sistema predstavlja povećanje pouzdanosti sistema. Distribuirani generatori u slučaju prestanka napajanja nekog dela potrošača u kome se oni nalaze mogu da nadomeste nedostatak električne energije dok se ne otkloni kvar i uspostavi normalno napajanje. U slučaju prekida napajanja potrošača prve kategorije, alternativni izvori napajanja mogu da se koriste kao rezervno napajanje takvih potrošača.

7. ZAKLJUČAK

U ovom radu predstavljene su koristi i problemi prilikom upotrebe distribuirane proizvodnje. Takođe, predstavljen je i veliki značaj korišćenja obnovljivih izvora energije, koji predstavljaju veliki neiskorišćeni potencijal.

Od distributivne proizvodnje se očekuje da ima veoma važnu ulogu u budućim elektroenergetskim sistemima. Međutim, jedan od problema distribuiranih izvora je svakako kompleksnost povezivanja, upravljanja i nadzora tih izvora u sistemu.

Rešenje ovog problema jeste upotreba pametnih mreža. To je elektroenergetski sistem budućnosti, čiji će elementi biti „pametni" i u mogućnosti da međusobno komuniciraju putem jedinstvene komunikacione mreže. Prednost pametnih mreža se ogleda u povećanju pouzdanosti, bezbednosti, sigurnosti sistema kao i upravljanju svim uređajima u mreži, između ostalog i distribuiranim izvorima.

Distribuirana proizvodnja bi trebalo u skorijoj budućnosti da imaju dominantni udeo udeo proizvodnji energije.

8. LITERATURA

[1] European Commission, „*EUROPE 2020: A strategy for smart, sustainable and inclusive growth*", 2010.

[2] Deloitte, *The future of the global power sector – Preparing for emerging opportunities and threats*

[3] Nenad Đajić, *Obnovljivi izvori – stanje i razvoj u Srbiji*, dostupno na http://www.energoportal.info/stranice/obnovljivi%20izvori/OIE%20-%20STANJE%20I%20RAZVOJ.pdf

[4] National Energy Technology Laboratory, „*Understanding the Benefits of the Smart Grid:Smart Grid Implementation Strategy*", U.S. Department of energy, 2010.

[5] *Renewables 2019, Global status repor*, dostupno na https://www.ren21.net/why-is-renewable-energy-important/

[6] Ioanna Xyngi: An Intelligent Algorithm for Smart Grid Protection Applications, Delft University of Technology, Delft, Netherlands

Kratka biografija:

Isidora Savić rođena je u Novom Sadu 1995. godine. Osnovne studije završila je na Fakultetu tehničkih nauka 2018. godine iz oblasti Elektrotehnike i računarstva – Elektroenergetski sistemi. Master rad je odbranila 2019. godine, na istom fakultetu, na smeru Energetika, elektronika i telekomunikacije – Elektroenergetski sistemi.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.38:620.9
DOI: https://doi.org/10.24867/07BE46Radivojkov

ANALIZA REZULTATA MERENJA PROPADA NAPONA U DISTRIBUTIVNIM MREŽAMA

ANALYSIS OF THE MEASUREMENTS RESULTS FOR VOLTAGE SAGS IN DISTRIBUTION GRIDS

Ivana Radivojkov, Vladimir Katić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – Energetika, elektronika i telekomunikacije

Kratak sadržaj – *Predmet rada jeste analiza merenja propada napona u savremenim distributivnim mrežama. Signali napona mereni su u realnoj evropskoj distributivnoj mreži na različitim mestima na dva naponska nivoa – 630 V i 1000 V. Obrađena je svojevrsna klasifikacija ovih signala, a zatim je data kvantitativna analiza dubine i trajanja propada napona, kao i maksimalne vrednosti i trajanja Harmonijskog otiska. Zatim je data statistička obrada rezultata do kojih se došlo kvantitativnom analizom.*

Ključne reči: *Kvarovi, Propadi napona, Distributivna mreža, Statistička analiza, Harmonici*

Abstract – *The subject of this paper is to analysis of the measurements results for voltage sags in modern distribution grids. Voltage signals were measured in a real European distribution grid at different locations at two voltage levels - 630 V and 1000 V. A sort of classification of these signals was performed, followed by a quantitative analysis of the depth and duration of the voltage sag, as well as the maximum value and duration of the Harmonic footprint. The results of the quantitative analysis were then statistically analyzed.*

Keywords: *Faults, Power quality, Voltage sags, Distribution grid, Statistical analysis, Harmonics*

1. UVOD

Elektroenergetski sistem (EES) čine četiri osnovna podsistema: proizvodnja, prenos, distribucija i potrošnja. U ovom radu predmet istraživanja su kvarovi koji se dešavaju isključivo u pojedinim delovima distributivnog podsistema, tj. distributivne mreže. Elektrodistributivne mreže predstavljaju deo elektroenergetskih sistema čija je funkcija da vrše raspodelu, odnosno distribuciju električne energije od napojnih čvorova koji se nalaze u transformatorskim stanicama visoki napon (VN)/srednji napon (SN) (koje se napajaju iz prenosnih ili subprenosnih mreža) do krajnjih potrošača električne energije. Tradicionalne distributivne mreže su se sastojale isključivo od pasivnih elemenata: potrošača, nadzemnih vodova, kablova, transformatora, kondenzatorskih baterija itd. Električna energija tradicionalno se nije proizvodila u distributivnim mrežama.

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

Zbog navedenih uslova, tokovi snaga u distributivnim mrežama bili su usmereni isključivo od napojnih čvorova prema potrošačima električne energije. Međutim, tokom poslednjih nekoliko decenija opisani koncept distributivnih mreža značajno se menja. Naime, savremene distributivne mreže prolaze kroz niz transformacija i unapređenja [1]. Pre svega, ove promene obuhvataju:

- Uvođenje distribuiranih energetskih resursa (distribuiranih i obnovljivih izvora el. energije i distribuiranih skladišta el. energije);
- Povećanje stepena upravljive potrošnje;
- Povećanje stepena automatizacije distributivnih mreža putem uvođenja daljinski kontrolisanih prekidača i drugih uređaja lokalne automatike.

Kao posledica navedenih promena, pogon savremenih distributivnih mreža postaje znatno dinamičniji u odnosu na tradicionalne distributivne mreže i samim tim podložan brojnim pogonskim izazovima, kao što su: dvosmerni tokovi snaga, problemi sa stabilnošću u prelaznim režimima, višestruki izvori u režimima sa kvarom itd.

Cilj svakog EES-a je da električnu energiju proizvedenu u nekoj vrsti elektrane putem prenosne i distributivne mreže isporuči do krajnjih potrošača. Pritom je potrebno da potrošač besprekidno bude napajan i dobija električnu energiju određenog kvaliteta. Kvalitet električne energije utiče na kvalitet rada (proizvodnje materijalnih i nematerijalnih dobara) i kvalitet života. Pojam kvaliteta električne energije je složen jer pored korisničkog, ekološkog, i komercijalnog kvaliteta podrazumeva tehnički kvalitet kao sinonim za kvalitet električne energije koji obuhvata probleme pouzdanost i sigurnosti napajanja, ukupne stabilnosti rada sistema, brojne poremećaje kao i uzajamni odnos elektroenergetski sistem – potrošač i obrnuto.

Međutim, nije uvek moguće obezbediti besprekidno napajanje i to kvalitetnom električnom energijom, jer na pojedinim delovima elektroenergetskog sistema dolazi do kvarova koji su relativno česta pojava. Kvar je bilo koje stanje koje se ne može definisati kao normalno ili stacionarno stanje u mreži. Može se reći još i da je električni kvar odstupanje napona i struje od nominalnih vrednosti ili stanja. Kada dođe do kvara, to prouzrokuje preveliku struju, što oštećuje opremu i uređaje. Električni kvarovi u trofaznom elektroenergetskom sistemu uglavnom su klasifikovani u dve vrste - prekid i kratak spoj. Pored ovoga, mogu postojati i kombinacije (simultanih) kvarova, kao i kvarovi namotaja opreme.

Prekid nastaje kada se dogodi kvar na putu provođenja električne energije, odnosno kada dođe do fizičkog prekida provodnika. Kratak spoj je u IEC 60 909 standardu definisan kao nastanak (slučajno ili namerno) provodne veze relativno male otpornosti ili impedanse između dve ili više tačaka električnog kola koje su u normalnom stanju na različitim potencijalima. Kvarovi nastaju usled različitih uzroka kada dolazi do električnih i/ili mehaničkih oštećenja.

Najčešći uzroci kvarova u EES-u su: vremenske prilike, kvarovi opreme i slabljenje izolacije, ljudski faktor, ptice, druge životinje, preopterećenje provodnika, itd. Svaka komponenta sistema može da se ošteti ili pokvari usled električnih i/ili mehaničkih naprezanja. Otkrivanje i analiza kvarova neophodna je za odabir ili projektovanje odgovarajuće opreme rasklopnih uređaja, elektromehaničkih releja, prekidača i drugih zaštitnih uređaja.

2. KVALITET ELEKTRIČNE ENERGIJE

Za potrošače u elektrodistributivnim mrežama od izuzetne važnosti je kontantno snabdevanje, dovoljna raspoloživa snaga i stabilni parametri napona na priključnim sabirnicama, odnosno konstantna isporuka i kvalitet isporučene električne energije. Ovi parametri čine osnovne postavke stabilno rada neke mreže i potpadaju pod pojam kvaliteta električne energije [7].

Za istraživanja u ovom radu razmatrane su dve bitne karakteristike kvaliteta električne energije: viši harmonivi i porpadi napona.

2.1. Viši harmonici

Viši harmonici se definišu kao neželjene spektralne komponente izobličenog signala čija je frekvencija jednaka celobrojnom umnošku osnovne frekvencije. Na primer, u sistemu gde je frekvencija 50 Hz, drugi harmonik će se pojavljivati na 2x50 Hz = 100 Hz, treći na 3x50 Hz = 150 Hz, itd. Termin „harmonik“ prvi put spominju 1894. godine, Houston i Kennelly [2]. Harmonicima su se, najpre, bavili matematičari, među njima najpoznatiji je Furije (Fureov razvoj u red, Furijeova transformacija).

Ukupna harmonijska distorzija (THD, Total harmonic distortion) definiše se kao odnos sume zbira kvadrata svih harmonijskih komponenti (viših harmonika) u odnosu na osnovni (prvi) harmonik [4]:

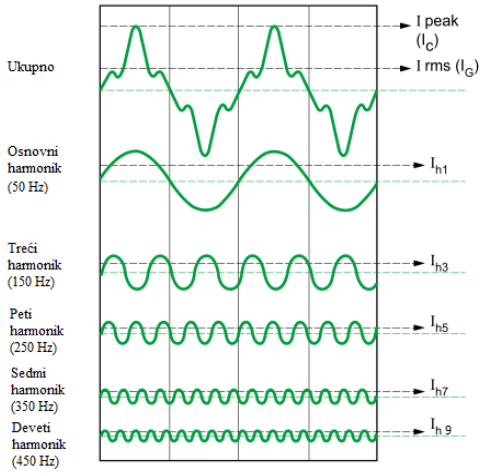
$$\mathrm{THD_U} = \frac{\sqrt{\Sigma_{h=2}^{\infty} U_h^2}}{U_1} \cdot 100\% \qquad (1)$$

Kako bi se ispitao udeo pojedinog hramonika u talasnom obliku napona, definiše se harmonijska distorzija napona (Harmonic distortion – HD) za harmonik h-tog reda kao:

$$\mathrm{HD_{U,h}} = \frac{U_h}{U_1} \cdot 100\% \qquad (2)$$

Na slici 1 prikazani su izobličeni talasni oblik struje i njegova raspodela na osnovni, 3-ći, 5-ti, 7-mi i 9-ti harmonik.

Viši harmonici izazivaju raznovrsne neželjene efekte. To su smetnje u radu računara, greške u radu zaštitne opreme, greške u merenju, prenaponi, interferencija sa komunikacionim i signalnim uređajima, povećanje gubitaka u generatoru, pregrevanje neutralnog provodnika, dodatno grejanje transformatora, probleme sa kondenzatorskim baterijama, probleme sa radom releja, neispravan rad merne opreme i dodatne gubitke prilikom transporta i distribucije el. energije, itd. [5].



Slika 1. *Izgled osnovnog, trećeg, petog, sedmog, devetog harmonika i ukupnog talsnog oblika struje* [3]

2.2 Propadi napona

Kao posledica kratkih spojeva, ali i drugih poremećaja u mreži, dolazi do pojave propada napona. Prema IEEE standardu 1159-2009 propadi napona su definisani kao redukcija napona u rasponu od 0.1-0.9 r.j. nominalne vrednosti napona, kada je frekvencija sistema nominalna, i kada je trajanje poremećaja u rasponu od pola periode do jednog minuta [6].

Propad napona je dvodimenzioni poremećaj, čiji nivo je određen kako dubinom propada napona (u odnosu na referentni ili nominalni napon), tako i njegovim trajanjem (vremenom). Na slici 2 predstavljen je izgled jednog propada napona i njegovi ključni parametri.

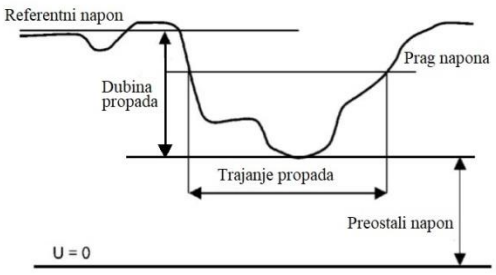
Referentni napon U_{ref} je vrednost napona navedena kao bazna vrednost u odnosu na koju se druge veličine, koje karakterišu poremećaj, izražavaju u relativnim jedinicama. U normalnom slučaju ovaj napon je jednak nazivnom naponu mreže U_n. Dubina propada definisana je kao razlika između najmanje efektivne vrednosti napona za vreme propada (preostalog napona) i referentnog napona U_{ref}. Prag napona je granična vrednost napona koja je dopuštena u mreži (obično ±10% U_n), odnosno 90% referentnog napona U_{ref}. [7].

Trajanje propada napona je vreme između trenutka u kojem napon u određenoj tački na sistemu napajanja padne ispod praga napona i trenutka u kojem poraste iznad praga napona.

Preostali napon je najmanja efektivna vrednost na koju opadne napon za vreme propada napona.

Kada se desi propad napona, izvori energije koji pod normalnim uslovima isporučuju energiju opremi ne obavljaju svoju funkciju ili je vrše samo u ograničenom rasponu. To dovodi do pogoršanja performansi opreme ili do potpunog prestanka rada. Posebno su primećeni problemi u radu računara, kompjuterske opreme u industriji, digitalnih industrijskih elektromotornih pogona i sličnih uređaja. Zaštitni sistemi se implementiraju u

svrhu isključenja napajanja kada napon padne ispod zadatog nivoa.



Slika 2. *Karakteristične veličine u propadu napona*

Međutim, tokom postojanja struje kvara, odnosno pojave propada napona u distributivnoj mreži, u talasnom obliku napona na sabirnici gde se pojavljuje ovaj propad, naglo porastu vrednosti viših harmonika, posebno 2-gog, 3-ćeg, 5-tog i 7-mog. To predstavlja svojevrsnu indikaciju da je došlo do ovakvog poremećaja.

U literaturi [8] ova pojava je predstavljena sa posebnog aspekta, odnosno posmatrana je ukupna vrednost seta ovih harmonika, koja se dobija kada se u izrazu (1) izračuna vrednost distorzije za h=2,3,5,7 (HDU2357). Pokazano je da ovakav parameter, nazvan Harmonijski otisak ima specifičan oblik čijom analizom se mogu dobiti informacije o nekim osobinama propada napona, odnosno samog kvara. Iz tog razloga izvršena je detaljna analiza raspoloživog skupa mernih rezultata.

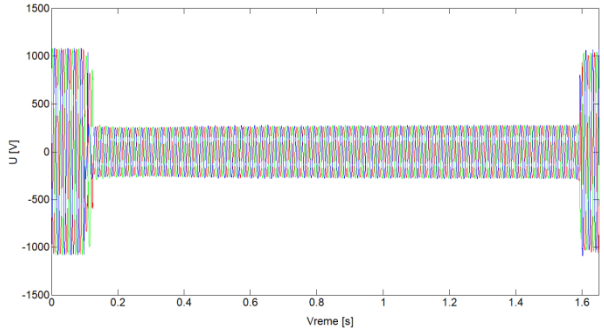
3. MERENI PODACI

Predmet istraživanja ovog master rada su signali napona mereni u jednoj realnoj evropskoj distributivnoj mreži na različitim mestima na dva naponska nivoa – 630V i 1000V (industrijski naponski nivoi). Analizirano je 106 merenih naponskih signala nakon neke vrste poremećaja u distributivnoj mreži. Na slici 3 predstavljen je primer jednog rezultata merenja. Mereno je na frekvenciji od 50 Hz, koja je standardna frekvencija u evropskim EES-ima. Rezolucija merenja je 12 bita. Broj odbiraka ili eng. „sample rate“ govori koliko brzo su odbirci uzimani i on je 96 kod svih merenih signala, što znači da je za svaki signal izmereno 96 tačaka u toku jedne periode.

Pošto su u ovom radu od interesa isključivo signali propada napona, a ne analiza kvarova, bez obzira koji se kvar u distributivnoj mreži dogodio, napravljena je sledeća klasifikacija merenih signala: grupa 1 – propadi napona u jednoj fazi (9.43%), grupa 2 – propadi napona u dve faze (27.36%), grupa 3 – propadi napona u tri faze (23.58%), grupa 4 – višestruki propadi napona (22.64%), grupa 5 – prekidi napajanja (11.32%), grupa 6 – magnećenje transformatora (1.89%), grupa 7 – tranzijenti (3.77%).

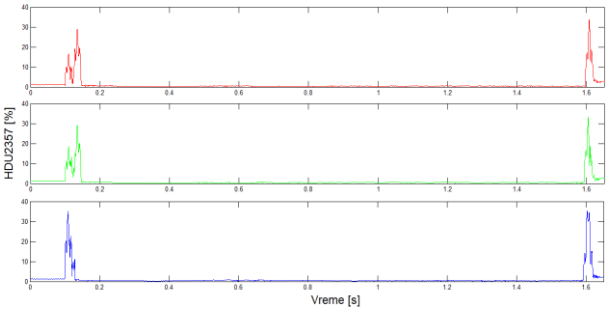
Kvantitativno su opisani mereni naponski signali, što podrazumeva analizu, pre svega, dubine propada i trajanja propada napona, koji predstavljaju glavne karakteristike bilo kog propada napona, a zatim su u istim signalima analizirani harmonici, odnosno formiran je Harmonijski otisak.

Harmonijski otisak predstavlja set od drugog, trećeg, petog i sedmog harmonika (HDU2357). Razlog zašto je izabran da se drugom i trećem harmoniku dodaju i peti i sedmi je da bi se dobila veća opštost i da bi svi kvarovi koji mogu da izazovu propade i poremećaje napona bili uspešno detektovani i prepoznati. Harmonici nultog reda, kao što je treći harmonik, se ne prenose kroz većinu distributivnih transformatora.



Slika 3. *Primer mernog naponskog signala – propad napona u tri faze*

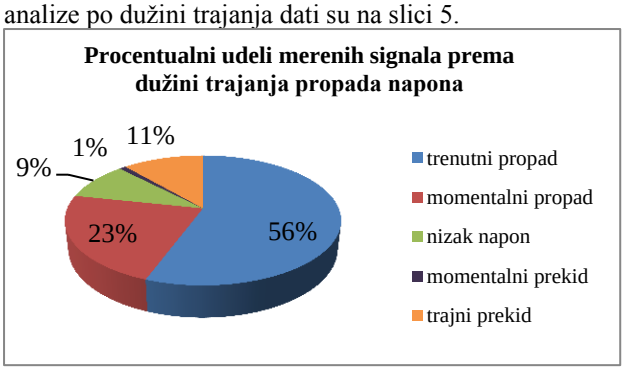
Na slici 4 prikazan je primer izgleda Harmonijskog otiska za trofazni propad. Dalja analiza obuhvatila je određivanje parametara Harmonijskog otiska, visina prvog i drugog pika.



Slika 4. *HDU2357 u sve tri faze mernog naponskog signala sa slike 3.1*

4. STATISTIČKA ANALIZA

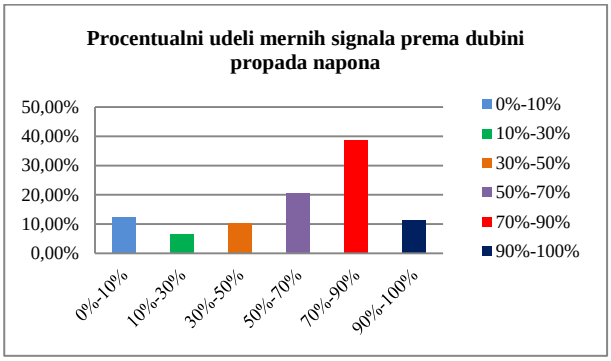
Propadi napona mogu se klasifikovati na nekoliko načina, a jedan od njih je prema trajanju, koja je data standardom IEEE 1159 o naponskim poremećajima. Ako je napon opao na vrednost između 10% i 90% nazivnog napona, to je propad napona, dok ukoliko je opao na vrednost nižu od 10%, to se tretira kao jedna od vrsta prekida. Rezultati analize po dužini trajanja dati su na slici 5.



Slika 5. *Procentualni udeli obrađenih mernih signala prema trajanju propada napona*

Analizom rezultata trajanja propada napona došlo se do sledećih brojčanih, odnosno procentualnih podataka: trenutni propadi (0.01s – 0.6 s) – 55.66%, momentalni

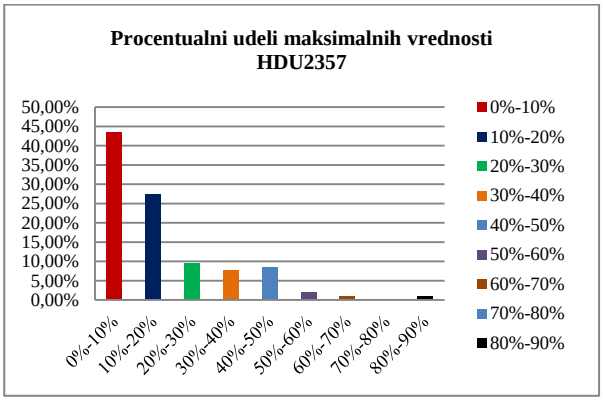
propadi (0.6 s – 3 s) – 22.64%, privremeni propad (3 s – 60 s) – 0%, nizak napon (60 s –) – 9.43%, momentalni prekid (0.01 s -3 s) – 0.94%, privremeni prekid (3 s – 60 s) – 0 %, trajan prekid (60 s –) – 11.32 %.



Slika 6. *Procentualni udeli obrađenih mernih signala prema dubini propada napona*

Tabela 1 *Prosečna, minimalna i maksimalna vrednost dubine i dužine propada napona*

	Prosečna vrednost	Minimalna vrednost	Maksimalna vrednost
Dubina propada napona	61.03%	95.10% (89.76%)	6.35% (17.23%)
Trajanje propada napona	0.45 s	0.02 s	1.56 s



Slika 7 *Udeo maksimalnih vrednosti Harmonijskog otiska*

Tabela 2 *Karakteristike Harmonijskog otiska*

	Prosečna vrednost	Minimalna vrednost	Maksimalna vrednost
Maks. vr. Harm. otiska	17.815%	2.45%	87.88%
Trajanje Harm. otiska	0.0345 s	0.0146 s	0.1442 s
Visina prvog pika	12.89%	2.05%	65.41%
Visina drugog pika	12.93%	1.95%	86.65%

5. ZAKLJUČAK

U ovom radu predstavljena je detaljna analiza opsežnog merenja različitih poremećaja u mreži. Takođe, urađena je i klasifikacija poremećaja, kao i proračun harmonika niskog spektra koji mogu biti upotrebljeni za detekciju, urađena njihova detaljna analiza i izračunavanje najbitnijih parametara kvara, kao i najbitnijih parametara Harmonijskog otiska.

Svi izračunati podaci su statistički obrađeni i izdvojeni u odgovarajuće kategorije, a zatim analizirani. Na osnovu dobijenih podataka može se zaključiti da je najveći broj poremećaja napona na niskom naponu uzrok kratkotrajnih propada napona, ukupno 79% procenata. Ovi poremećaji su dalje klasifikovani u dve podkategorije kao trenutni (56%) i momentalni propadi (23%). Trajnih kvarova je u ukupnom setu podataka svega 20%, i u velikom broju slučajeva se mogu klasifikovati kao trajni prekid (11%), dok ''nizak napon'' čini preostalih 9%. Kod prolaznih propada, izračunato je da je srednja vrednost trajanja 0.45s, dok je najkarći propad trajao svega 0.02s a najduži 1.56s. Za visinu Harmonijskog otiska, koja se može koristiti u savremenim metodama detekcije poremećaja iz izvedenog proračuna vidi se da je prosečna vrednost Harmonijskog otiska 17.815%, dok je prosečno trajanje ove pojave 0.0345s. Čak oko 71 % obrađenih uzoraka ima maksimalnu vrednost Harmonijskog otiska u opsegu od 0% do 20% vrednosti osnovnog harmonika, dok 25.48% uzoraka ima maksimalnu vrednost u opsegu od 20% do 50%, što znači da manje od 4% ima ovu vrednost koja je veća od 50% osnovnog harmonika

6. LITERATURA

[1] N. Kovački, Doktorska disertacija "Operativno planiranje rekonfiguracije distributivnih mreža primenom višekriterijumske optimizacije", 2017.

[2] G.K. Singh, "Power system harmonics research: a survey" 2007.

[3] http://www.aspap.org.

[4] A. P. Technologies, "Total Harmonic Distortion and Effects in Electrical Power System."

[5] V. Šinik, "Uticaj nelinearnih potrošača male snage na kvalitet električne energije", 2017.

[6] The Institute of Electrical and Electronics Engineers, *Recommended Practice for Monitoring Electric Power Quality. IEEE*. 1995.

[7] Ž. Novinc "*Kvalitet električne energije*", Elektrotehnički fakultet Osijek, 2006.

[8] V.A. Katic, A. Stanisavljevic, "Smart Detection of Voltage Dips Using Voltage Harmonics Footprint", *IEEE Trans. on Industry Application*, Vol.54, No.5, Sep./Oct. 2018, pp.5331-5342

Kratka biografija:

Ivana Radivojkov rođena je u Novom Sadu 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Elektroenergetski sistemi odbranila je 2019.god. kontakt: radivojkov.i@gmail.com

prof. dr Vladimir Katić rođen je u Novom Sadu 1954. god. Doktorirao je na Fakultetu tehničkih nauka 1991. god., a od 2002. je zvanju redovni profesor. Oblast interesovanja su energetska elektronika, električna vozila, obnovljivi izvori i kvalitet električne energije.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE47Stanisic

СМЕРНИЦЕ ЗА РАЗВОЈ КОРИСНИЧКОГ ИНТЕРФЕЈСА ЗА НАДЗОР И КОНТРОЛУ ПМИС СИСТЕМА

GUIDELINES FOR THE DEVELOPMENT OF A USER INTERFACE FOR MONITORING AND CONTROL OF RWIS SYSTEM

Душан Станишић, *Факултет техничких наука, Нови Сад*

Област – РАЧУНАРСТВО И АУТИМАТИКА

Кратак садржај – *У раду је разматран проблем интеракције диспечера зимског одржавања са ПМИС системом у циљу скраћења времена реаговања зимске службе. Проблем је решен применом технологија из области „интеракција човек-рачунар" и принципа системског инжењерства и инжењерства употребљивости. У раду је доказана истраживачка хипотеза:*
„Унапређењем корисничког интерфејса путног метеоролошко-информационог система повећаће се ефикасност и смањити трошкови зимског одржавања путева."
Резултати истраживања преточени су у Смернице за развој корисничког интерфејса за надзор и контролу ПМИС система.
Кључне речи: *интеракција човек-рачунар, кориснички интерфејс, путни метеоролошки информациони систем*
Abstract – *The paper analysed the issue of the winter maintenance dispatcher interacting with the RWIS system, with the aim of reducing winter services response time.*
The problem was solved by applying Human-Computer Interaction technologies and principles of systems engineering and usability engineering. The paper proved the research hypothesis:
"Improving the user interface of the Road Weather Information System will improve efficiency and reduce costs of winter road maintenance."
The research results have been translated into the Guidelines for the Development of a User Interface for Monitoring and Control of RWIS system.
Keywords: *Human-Computer Interaction, User Interface, Road Weather Information System.*

1. УВОД

Зимско одржавање путева по моделу Уговора на бази квалитета пружене услуге (енгл. *winter maintenance performance-based contracts*) најсавременије је достигнуће у области зимског одржавања путева које остварује високе уштеде и унапређује сам процес одржавања [1,2].

Предуслов за реализацију Уговора о зимском одржавању на бази квалитета пружене услуге јесте функционалан путни метеоролошки информациони систем, како би услуга зимског одржавања уговорена по овом моделу била дефинисана између уговорних страна квантитативно и квалитативно [1] (енг. *RWIS Road Weather Information System*). По овом моделу зимског одржавања остварују се уштеде и до 47% [1].

Путни метеоролошки информациони систем се дефинише као скуп технологија које користе историјске и тренутне метеоролошке податке како би се формирала довољна количина информација о временским условима на путу на основу којих би се иницирале исправне и правовремене корективне и превентивне мере зимског одржавања [1].

Стога, интеракција коју диспечер зимског одржавања врши са путним метеоролошким информационим системом путем графичког корисничког интерфејса је кључна за процес зимског одржавања путева. Путни метеоролошки информациони систем, као такав, спада у „системе за подршку при доношењу одлука" (енг. *Decision Support Systems*).

Смањење времена реаговања зимске службе применом технологија из области интеракција човек-рачунар доказ је да технологије и знања из области интеракција човек-рачунар проналазе успешну примену у великом броју привредних сектора, а не само, као што је устаљено мишљење, у индустрији забаве, електронским медијима и индустрији рачунарских игара.

У складу са овом тврдњом формирана је и истраживачка хипотеза која гласи: „Унапређењем корисничког интерфејса путног метеоролошко-информационог система повећаће се ефикасност и смањити трошкови зимског одржавања путева."

2. МЕТОДЕ ИСТРАЖИВАЊА

Научни приступ у проучавању области интеракција човек рачунар почео је применом метода експерименталне психологије и комбинованих метода за прикупљање података уз свеобухватну подршку моћних софтверских алата. Убрзо је уочено да се и традиционалне научне методе могу успешно примењивати у проучавању области интеракција човек рачунар [3].

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Драган Иветић, ред. проф.

Смернице за развој корисничког интерфејса за надзор и контролу путног метеоролошко информационог система израђене су применом:

- Шнајдерманове методе,
- Шнајдерманових златних правила за израду корисничког интерфејса,
- принципа системског инжењерства,
- принципа инжењеринга употребљивости,
- практичног знања из области информационих технологија и
- практичног знања из области ПМИС система.

Примена практичних знања из области информационих технологија и ПМИС система је неопходна јер се у пракси показало да принципи и смернице за развој корисничког интерфејса првенствено настају из практичног искуства и емпиријских студија [3].
Овакав приступ решавању проблема усвојен је из разлога јер је интеракција човек-рачунар (енг. *Human-computer interaction HCI*) првенствено интердисциплинарна област која обухвата изучавање, пројектовање и креирање интеракције између људи и рачунарских система а налази се негде у пресеку друштвених и бихевиористичких наука са рачунарским и информационим технологијама.
Како је интеракција човек-рачунар једна од најбрже растућих области која све више добија на значају [4] то су и методе за њено изучавање у непрекидном развоју.

3. СМЕРНИЦЕ ЗА РАЗВОЈ КОРИСНИЧКОГ ИНТЕРФЕЈСА ЗА НАДЗОР И КОНТРОЛУ ПМИС СИСТЕМА

Полазна смерница зимског одржавања је: „Посути праву количину материјала на право место у право време”.

Развојне смернице које следе настале су на бази мера за унапређење ПМИС система које су резултат примене технологија из области „интеракција човек рачунар“ на проблеме идентификоване анализом постојећег стања у циљу њиховог ублажавања:

1) Путем графичког корисничког интерфејса у циљу елиминације нагађања, како и када да реагује, адекватно и благовремено информисати диспечера зимског одржавања о метеоролошкој и саобраћајној ситуацији која влада на и у околини територије одржавања. У том циљу путем интерактивних мапа диспечеру приказати: метеоролошке податке, саобраћајне податке, временске прогнозе, упозорења о очекиваним екстремним временским условима и сл.

2) Графички кориснички интерфејс ПМИС система пројектовати и израдити у складу са Шнајдермановим златним правилима за израду корисничког интерфејса:

a. тежити конзистентности,
b. обезбедити пречице искусним корисницима,
c. пружити повратне информације,
d. у корисничком дијалогу понудити и акцију којом се окончава текући задатак,
e. омогућити једноставно управљање грешкама,
f. пружити могућност повратка уназад - дозволити опозив операција,
g. пружити корисницима осећај контроле,
h. свести оптерећење краткорочне меморије на најмању могућу меру – смањити количину информација које је потребно краткорочно памтити.

3) Приликом пројектовања корисничког интерфејса, узети у обзир карактеристике физичког окружења у ком се одвија интеракција човека и рачунара као и карактеристике хардверских уређаја који се користе у комуникацији са диспечером зимског одржавања.

4) Графички кориснички интерфејс ПМИС система пројектовати тако да се извршава на:

a. радној станици главног инжењера ПМИС система,
b. радној станици диспечера зимског одржавања,
c. преносним уређајима извршилаца одржавања,
d. радној станици сервисера.

Изглед графичког корисничког интерфејса прилагодити зависно од типа уређаја на ком се извршава.

5) Графички кориснички интерфејс пројектовати тако да од презентације информација кориснику на основу којих се од корисника очекује да донесе одлуку, па до издавања наредбе за извршавање даљих радњи, мора да прође две секунде (нпр. деактивирати могућност уноса на две секунде). У те две секунде кориснику јасно мора да буде предочено да није у могућности да изда наредбу (нпр. деактивирано „засивљено” дугме.

6) Током процеса одлучивања кориснику истовремено презентовати предлог даљих мера зимског одржавања и пет до девет информација на основу којих су предложене те мере.

7) Предлози мера које систем предлаже кориснику морају да буду засновани на:

a. оперативним плановима насталим на бази одредби Уговора о зимском одржавању по моделу квалитета пружене услуге,
b. правилима струке и
c. захтевима корисника.

Оперативни планови морају да буду одобрени од стране инвеститора и да се састоје од описа инцидента и мера за његово ублажавање. Потребно је напоменути да оперативним плановима извођач прихвата ризике који су укључени у фиксну цену одржавања уз могућу опасност од појаве превида.

8) Предлог даљих мера у циљу ублажавања последица инцидента (тј сценарија) презентовати са кратким текстуалним описом и алгоритамским приказом даљих корака, заједно са пет до девет информација на основу којих је систем предложио тај предлог мера. Интеракцију са корисником реализовати као „мени тока”.

9) По покретању активности зимског одржавања на корисничком интерфејсу приказати: опис и статус планираних активности и активности које су у току, време у ком извођач мора да изврши корективну радњу (уговорено време реаговања), износ умањења цене услуге уколико иста није извршена у уговореном року и податак о категорији саобраћајнице која се одржава.

10) Уколико територија одржавања није покривена довољним бројем путних метеоролошких станица, тај податак презентовати оператеру јер директно утиче на тачност детекције инцидента.

11) Како би корисник у сваком тренутку знао у ком радном режиму се налази ПМИС систем (редован, инцидентни, режим одржавања) сваки од основних приказа мора да има преовлађујућу боју другачију од друга два мода (рецимо да се боја *Title Bar*-а разликује за свако стање система).

12) Време одзива графичког корисничког интерфејса не сме да буде дуже од 15 секунди. За то време ставити кориснику до знања да је његова инструкција примљена и да се његов захтев процесира.

13) Диспечера зимског одржавања путем графичког корисничког интерфејса обавештавати (алармирати) и визуелним и аудио путем.

14) Графички кориснички интерфејс мора да има могућност приказа табеле аларма која садржи основне информације о алармима:

a. „тежина аларма“ (информација, упозорење, аларм, опасност)

b. време појаве,

c. време потврде аларма,

d. објекат (*datapoint*) под алармом,

e. опис аларма,

f. вредности параметара које су довеле до појаве аларма

15) За сваки аларм из табеле аларма графички кориснички интерфејс мора да буде у стању да на захтев корисника прикаже предлог мера за ублажавање последица инцидента који је проузроковао аларм (алгоритам одржавања).

16) Графички кориснички интерфејс путног метеоролошког информационог система пројектовати тако да има могућност приказа на видео зиду. Водити рачуна о томе да WIMP елементи корисничког интерфејса нису погодни за приказ на видео-зиду (енг. *Windows, Icons, Menus, Pointer*)

Једна од мера је да се користе приручни менији који могу бити иницирани било где тако да не заклоне велики део приказа на видео-зиду. Оваквим приступом корисник неће морати да се враћа на палету алатки за даљи одабир.

17) WIMP начин интеракције свести на најмању могућу меру како би корисник био фокусиран на решавање задатка а не на начин руковања корисничким интерфејсом.

18) Прорачунати оптималну удаљеност корисника од видео зида. Прилагодити прорачун расвете просторије видео-зиду у циљу спречавања нежељеног одсјаја.

19) Издвојити три монитора из конфигурације видео зида који би имали функцију „алармних монитора”.

20) Графички кориснички интерфејс путног метеоролошког информационог система мора да буде у могућности да врши приказ просторних података и интерактивних мапа са свим за такав вид приказа, подразумеваним функцијама (зумирање, „пановање” и сл.).

Просторни подаци од значаја за зимско одржавање путева су локације: мостова, усека, тунела, места на којима по искуству радника са терена долази до честе појаве леда, локације путних метеоролошких станица и др.

21) Кориснички интерфејс ПМИС система мора да поседује могућност једноставног додавања нових ентитета (проширивост) као што је рецимо додавање нове путне метеоролошке станице.

22) Надзирати и приказивати статусне податке свих елемената ПМИС система (укључујући и серверски део) у циљу очувања функционалности система.

23) Пратити статус: *web* сервера, RDBMS сервера, апликативног сервера, заштитног firewall-а, телекомуникационе опреме, система беспрекидног напајања, статус путних метеоролошких станица и статусне податке других уређаја. По детектованој неправилности обавестити сервисерану најкраћем могућем року.

24) Благовремено обавештавати сервисера о планираним превентивним мерама одржавања.

25) Благовремено обавештавати диспечера зимског одржавања о планираним превентивним мерама зимског одржавања.

26) Пратити, приказивати и обрађивати информације добијене од екстерних система као што су Републички хидрометеоролошки завод, АМСС, контролни центри, медијске организације и др.

27) Кориснички интерфејс мора да има могућност симултаног приказа више видео-стримова са видео-надзорних камера инсталираних на путним метеоролошким станицама.

28) Омогућити руковање мултимедијалним подацима: снимање, репродукција, претрага, архивирање и сл.

29) Пратити дешавања и бити активан на друштвеним мрежама у циљу размене информација о стању на терену.

30) Путем графичког корисничког интерфејса ПМИС система, пратити националне и регионалне електронске информативне сервисе.

31) Пратити информације о најавама прекида екстерних сервиса: искључења електричне енергије, планираним прекидима интернет саобраћаја и сл.

32) Омогућити креирање извештаја путем графичког корисничког извештаја. Изглед и садржај извештаја усагласити са инвеститором.

33) У циљу обуке и периодичне провере корисника, омогућити симулацију ванредних ситуација путем графичког корисничког интерфејса.

34) Права приступа ПМИС систему контролисати аутентикацијом и ауторизацијом корисника.

35) Путем графичког корисничког интерфејса обезбедити приступ техничкој подршци и помоћи корисницима. Обезбедити локална и *on-line* корисничка упутства, *on-line* техничку помоћ и слично.

36) У циљу помоћи диспечеру зимског одржавања, омогућити приказ: законске регулативе из предметних области (законе и правилнике), податке информативног центра ЈППС, извештаје о стању на путевима, извештаје о временским приликама, извештаје са граничних прелаза и електронско интерактивно упутство ПМИС система.

4. ЗАКЉУЧАК

Време реаговања зимске службе кључан је параметар зимског одржавања и као такав један је од одлучујућих показатеља учинка извођача.

Неадекватне и/или неблаговремене мере зимског одржавања главни су узрочници појаве негативних социолошких, еколошких и економских утицаја на становништво.

Јачина негативног утицаја нежељених појава директно је сразмерна времену које протекне од појаве инцидента па до успостављања задовољавајућег нивоа услуге саобраћајнице. Истраживањем је показано да значајну улогу у времену реаговања зимске службе имају:

- време потребно за обавештавање извршилаца и диспечера зимског одржавања и
- време потребно за доношење правовремених исправних одлука.

Неадекватна времена одзива графичког корисничког интерфејса (и сувише кратка и сувише дугачка) у већини случајева доприносе повећању:

- ризика од доношења неблаговремене одлуке диспечера зимског одржавања
- ризика од доношења погрешне одлуке диспечера зимског одржавања
- ризика од неадекватног времена одзива (кашњења) извршилаца зимског одржавања
- ризика од доношења неблаговремене одлуке тима за управљање кризним ситуацијама

Поред неадекватног времена одзива на појаву наведених ризика значајан утицај врши и начин обавештавања диспечера, начин презентације података тј. начин на који диспечер зимског одржавања путем графичког корисничког интерфејса врши интеракцију са ПМИС системом.

Управо у циљу оптимизације времена одзива графичког корисничког интерфејса и унапређења помоћи диспечеру у процесу доношења одлука, примењене су технологије и знања из области „интеракција човек-рачунар".

Оваквим приступом потврђена је хипотеза да: „Унапређење корисничког интерфејса путног метеоролошко-информационог система повећава ефикасност и смањује трошкове зимског одржавања путева."

5. ЛИТЕРАТУРА

[1] Ненад Аћимовић, „Улога и функција путног метеоролошког информационог система (RWIS) у редовном одржавању путева у републици Србији", Зборник радова прве међународне научно-стручне конференције о савременом одржавању путева, стр. 167-174, новембар 2013.

[2] др Игор Јокановић, „Праћење извршења уговора о одржавању путева према дефинисаном нивоу услуге", Стручни рад УДК: 625.7.08 ; 351.712.2:625.7/.8 = 861, Грађевински материјали и конструкције, вол. 54, бр. 4, стр. 7-24, 2011.

[3] B. Shneiderman & C. Plaisant, Дизајнирање корисничког интерфејса. Београд: ЦЕТ, 2006.

[4] Hewett, Baecker, Card, Carey, Gasen, Mantei, Perlman, Strong and Verplank; ACM SIGCHI Curricula for Human-Computer Interaction.

[5] Драган Иветић, Формална спецификација корисничког интерфејса интерактивног графичког система, докторска дисертација, Нови Сад, Факултет техничких наука, 1999.

Кратка биографија:

Душан Станишић рођен је у Београду 1972. године. Мастер рад на Факултету техничких наука из области Електротехнике и рачунарства – рачунарство и аутоматика одбранио је 2019. године. Више од двадесет година област интересовања му је примена информационих технологија у путној привреди.
контакт: dusan.stanisic@lineal.rs

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 007.51
DOI: https://doi.org/10.24867/07BE48Trkulja

IMPLEMENTACIJA I TESTIRANJE ALGORITAMA ZA TAČNO PODUDARANJE STRINGOVA

IMPLEMENTATION AND TESTING OF STRING MATCHING ALGORITHMS

Branislav Trkulja, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu su opisani algoritmi koji su najčešće u upotrebi za tačno podudaranje stringova. Razmatrani su "Brute force" metoda, algoritam Rabin-Karp, Knuth-Morris-Pratt i Boyer Moore algoritam. Nabrojani algoritmi su implementirani u C# programskom jeziku i izvršena su testiranja njihovih performasi. Rad sadrži rezultate dobijene testiranjem nad dve vrste testnih podataka. Jedna vrsta podataka su veštački generisani primeri koji treba da istaknu dobre i loše strane algoritama, dok drugu vrstu podataka čine realni primeri: tekst knjige „Rat i mir" i CIM/XML fajl sa 2GB podataka.*

Ključne reči: *Pretraga teksta, Knuth-Morris-Pratt algoritam, Boyer Moore algoritam, Rabin Karp algoritam*

Abstract – *This paper describes the most commonly used algorithms for exact string matching. The Brute force method, the Rabin-Karp algorithm, the Knuth-Morris-Pratt algorithm and the Boyer Moore algorithm are discussed. The above algorithms were implemented in C # programming language and their performance was tested. The paper contains the results obtained by testing over two types of test data. One type of data is artificially generated examples that should highlight the good and bad sides of algorithms, while the other type of data is real examples: a 3226571 character text book and CIM / XML file with 2GB data.*

Keywords: *Text searching algorithms, Knuth-Morris-Pratt algorithm, Boyer Moore algorithm, Rabin Karp algorithm*

1. UVOD

U računarstvu algoritmi podudaranja stringova su važna klasa string algoritama koji pokušavaju da nađu mesto (ili više mesta) na kome se unutar teksta nalazi traženi obrazac.

Tipično, obrazac i tekst se sastoje od niza karaktera, gde tekst može biti ceo dokument, a obrazac može biti jedna reč, njen deo ili tekstualni fragment. Obično je tekst nastao iz nekog prirodnog jezika, ali to može biti bilo koja vrsta binarnih podataka u računaru.

Kada se takav binaran podatak pretražuje mogu se primeniti algoritmi podudaranja stringova, čime oni dobijaju više na značaju.

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

Problem sa pretragom teksta jeste pronaći svako podudaranje obrasca u velikom tekstu. Podudaranje predstavlja da su svi karakteri obrasca sadržani u tekstu u istom redosledu. Ovaj problem se jasno javio u sedamdesetim godinama prošlog veka, kada su kompanije koje programiraju počele proizvoditi softversku obradu teksta i povećale potrebu za pronalaženjem brzih i efikasnih rešenja za ovaj problem. Takođe je jasno da je potraga za tekstom osnova postupka pronalaženja informacija u mnogim poljima, kao što su baze podataka, pretrage sadržaja knjiga, kao i pretraga web sadržaja.

Postojeći algoritmi pretrage su razvrstani u dve kategorije: algoritmi tačnog podudaranja i algoritmi približnog podudaranja. Neki od algoritama tačnog podudaranja su *Brute force*, Knuth-Morris-Pratt, Boyer-Moore i Rabin Karp algoritam. Algoritami približnog podudaranja omogućavaju da odredimo ograničen broj različitih znakova u podudaranjima.

Ovaj rad se bavi gore nabrojanim algoritama tačnog podudaranja i ima za cilj da testira i pokaže primenu tih algoritama za pretragu velikih tekstualnih fajlova na osnovu unetog obrasca. Takođe, analizirana je složenost algoritama i rad pokušava da pomogne pri izboru algoritma za odredjenu pretragu.

U poglavlju 2. će biti opisani poznati algoritmi za pretragu. U poglavlju 3. će biti prikazani rezultati testiranja algoritama za poduaranje obrazaca.

2. POZNATI ALGORITMI

2.1. Brute force algoritam

Brute force (BF) je metoda gde se koristi „gruba sila" da bi se dobilo jednostavno rešenje. Ona se sastoji od proveravanja svih pozicija u tekstu: da li se početak obrasca (prvi karakter) nalazi tu ili ne? Algoritam ne zahteva pretprocesiranje, i poredi obrazac sa tekstom s leva na desno [1]. Nakon uspešne provere prvog karaktera, obrazac i tekst se pomeraju za tačno jednu poziciju i opet se vrši provera jednakosti – sada drugih karaktera po redu. U slučaju da se karakteri ne podudaraju, prelazi se na proveru sledećeg karaktera teksta (prvi sledeći karakter u odnosu na prvo slovo prethodno proveravanog teksta) i prvog karaktera obrasca. U slučaju da se karakteri podudaraju nastavlja se sa poređenjem narednih karaktera u obrascu i tekstu i tako redom sve do kraja obrasca. Pomeranje se uvek vrši za tačno jedan karakter, zato što algoritam nema nikakve mehanizme koji bi pomogli da se odredi dužina pomeraja.

Što se tiče efikasnosti, u najgorem slučaju broj poređenja je $O(nm)$, gde su n i m dužine teksta i traženog obrasca. Međutim, ako postoje i u tekstu i u obrascu isti karakteri koji se ponavljaju, tada je složenost algoritma $O(m(n-m+1))$. Na primer, ako je obrazac „aaa“, a tekst „aaaaaaaaaa“, tada su $m = 3$ i $n = 10$, i broj poređenja je 24 [4]. Najbolji slučaj je kada se prvi karakter obrasca uopste ne nalazi u tekstu koji se pretražuje i tada je broj poređenja $\Omega(n)$.

2.2. Rabin-Karp algoritam

Rabin-Karp (RK) algoritam koristi heširanje da pronađe traženi obrazac u tekstu. On pokušava da ubrza testiranje podudaranja obrasca i dela teksta koji se trenutno proverava upotrebom heš funkcije. Heš funkcija je funkcija koja pretvara obrazac ili deo teksta koji se proverava u numeričku vrednost, koja se zove heš vrednost. Algoritam koristi činjenicu da ukoliko su obrazac i deo teksta koji se proverava jednaki, onda je jednaka i njihova heš vrednost. Jedan od potencijalnih problema jeste što postoje jednake heš vrednosti za različite obrasce. To znači da ako su heš vrednosti jednake, ne mora da znači da se i obrazac i tekst podudaraju, već se mora izvršiti dodatno proveravanje - karakter po karakter. Glavna ideja ovog algoritma je efikasno izračunavanje heš vrednosti za delove teksta dužine obrasca i za to se koristi *rolling hash* funkcija.
Rabin-Karp algoritam koristi $O(m)$ vremena za pretprocesiranje, dok je u najgorem slučaju vremenska složenost algoritma $O((n-m+1)m)$ [7].

2.3. Knuth-Morris-Pratt algoritam

Knuth-Morris-Pratt (KMP) algoritam je dosta sličan *brute force* metodi, ali najveća razlika je u tome što se, iterator koji se kreće kroz tekst, nikada ne vraća u nazad. Osnovna ideja algoritma je da se u trenutku kada se otkrije neusaglašenost teksta i obrasca (posle podudaranja nekoliko karaktera), već znamo neke od znakova u tekstu sledećeg prozora (deo teksta za koji se trenutno vrši provera o podudaranju). Ta informacija se upotrebljava da se izbegne provera znakova za koje se već zna da će se ionako podudarati.
Algoritam se sastoji iz dva koraka: pretprocesiranje i samo izvršavanje pretrage teksta. U pretprocesiranju se obrađuje sam obrazac. Uz pomoć njega se konstruiše pomoćni niz veličine m (iste veličine kao i obrazac), koji se koristi za preskakanje znakova koji se podudaraju.
Kada se naiđe na neslaganje teksta sa karakterom obrasca, tabela iz pretprocesiranja određuje sa kojim karakterom obrasca treba nastaviti poređenje, a da pri tom pozicija (pokazivač) u tekstu ostane na istom mestu. Ukoliko se indeks, koji pokazuje na obrazac, postavi na 0, to znači da se prvi karakter obrasca i trenutni karakter teksta ne podudaraju, i u tom slučaju se pokazivač teksta pomera za jedno mesto unapred. Ovo znači da se u svakom prolazu pomera ili pokazivač na tekst ili pokazivač na obrazac.
Što se tiče efikasnosti, algoritam KMP ispituje svaki karakter teksta tačno jednom. Vreme rada funkcije koja obrađuje obrazac i od njega pravi pomoćnu tabelu je $O(m)$, što se određuje metodom amortizovane analize. Slična amortizovana analiza, je korišćena i kod ispitivanja složenosti same pretrage. Vreme pretrage je $O(n)$. Pošto se algoritam sastoji iz dva dela, složenost celokupnog KMP algoritma je $O(n+m)$ [3].

Sam algoritam ni nema neku unapređenu verziju sebe. Način za ubrzavanje vremena izvršavanja algoritma jeste paralelizacija. Koncept paralelizacije je uveden radi poboljšanja performansi algoritma. Koristeći koncept paralelizacije, niz veoma dugačkog teksta je podeljen na delove nezavisno od veličine obrasca. Isti obrazac se izvršava na različitim delovima teksta paralelno, smanjujući vreme izvršavanja algoritma. Ovde se KMP algoritam primenjuje na zasebne delove teksta u paraleli. Glavni problem paralelizacije jeste da ako se podudaranje obrasca i teksta nalazi tačno da delu podele podataka ili tački veze. Za rešavanje ovog problema na svakoj tački povezivanja vrši se dodatna obrada podataka. Ukoliko je dužina obrasca m, uzima se $m-1$ karakter od kraja prvog dela, i $m-1$ karakter od početka drugog dela i spoje u jedan string, koji se naknadno proveravaju uz pomoć KMP algoritma.

2.4. Boyer-Moore algoritam

Za razliku od predhodnih algoritama, Boyer-Moore (BM) sadrži tri pametne ideje: skeniranje s desna na levo, „bad character shift rule“ i „good suffix shift rule“ [4, 5].

Boyer-Moore algoritam proverava obrasce skeniranjem karaktera s desna na levo, a ne s leva na desno kao u predhodnim algoritmima. Ideja *lošeg karaktera* je jednostavna. Znak teksta koji se ne podudara sa trenutnim karakterom obrasca naziva se „Bad Character“. Nakon neusklađenosti imamo dva moguća scenarija. Ukoliko se *loš karakter* nalazi u obrascu, u tom slučaju se „poravnava“ obrazac sa tekstom, tako što se poravnaju pokazivači *lošeg karaktera* i odgovarajućeg karaktera obrasca. Sa druge strane ukoliko se *loš karakter* ne nalazi u obrascu, tada se preskače tekst za celu dužinu obrasca. Kod ideje *dobrog sufiksa* posmatramo delove teksta *t* koji se podudaraju sa delovima obrasca. Imamo tri moguća scenarija. Obrazac može da sarži nekoliko istih pojava *t*. U tom slučaju pokušavamo da pomerimo obrazac tako da bi se pojava *t* poklopila sa tekstom. Drugi slučaj je da pokušamo da nadjemo sufiks od *t* koji će se podudariti sa prefiksom od obrasca. I poslednji slučaj je da nemamo uopšte poklapanja obrasca i dela teksta *t*, i tada samo pomerimo obrazac za celu dužinu unapred.

Boyer-Moore algoritam ima najgori slučaj izvođenja $O(n+m)$, ako se uzorak ne pojavljuje u tekstu. Vremenska složenost u fazi pretprocesiranja je $O(m+\delta)$ (δ je broj različitih znakova koji se mogu pojaviti u tekstu). Ukoliko se uzorak pojavljuje u tekstu, vreme izvršavanja algoritma je $O(nm)$ u najgorem slučaju [2, 6]. Iako je ovo ista složenost kao i vremenska složenost *brute force* algoritma, algoritam Boyer-Moore se u praksi bolje pokazuje zbog izostajanja nepotrebnih poređenja.

2.5. Boyer-Moore-Horspool algoritam

Boyer-Moore-Horspool algoritam uvodi tabelu *lošeg podudaranja*. Kada se znakovi ne podudaraju, pretraga skače na sledeće mesto podudaranja u obrascu pomoću tabele. Realizacija ovakvog rešenja može imati kraće vreme izvršavanja od mnogih drugih algoritama pretraživanja jer ne proverava sve znakove teksta, već preskače tekst uz pomoć tabele. Algoritam je brži što je obrazac duži. U najgorem slučaju, performanse algoritma su $O(nm)$, a u najboljem slučaju vreme je $\Omega(n/m)$ [6].

2.6. Boyer-Moore-Horspool-Sunday algoritam

Boyer-Moore-Horspool-Sunday algoritam proširuje ideju korišćenja *lošeg karaktera*. Prilikom ne podudaranja, proverava se sledeći karakter teksta da bi se odredio pomak. Ukoliko se taj karakter ne podudara ni sa jednim karakterom u obrascu, preskače se tekst za dužinu obrasca +1, u suprotnom se proveravani sledeći karakter poravnava sa prvim desnim identičnim karakterom obrasca. Najgora vremenska složenost je $O(nm)$, dok je najbolja vremenska složenost $\Omega(n/m+1)$. Algoritam je brži pri kraćim obrascima [6].

3. TESTIRANJE ALGORITAMA

Algoritmi BF, KMP, RK i BM su napisani u C# programskom jeziku, a zatim su izmerena vremena njihovog izvršavanja nad testnim podacima. Dobijeni rezultati su ovde prikazani i diskutovani.

U eksperimentu je upotrebljen PC računar sa Intel i3-4160 CPU na 3.60GHz i 32GB RAM-a. Operativni sistem je Windows 10 Enterprise 64-bitni.

Testiranje je izvršeno nad dve vrste podataka. Jedna vrsta podataka su veštački generisani primeri koji treba da istaknu dobre i loše strane algoritama, dok drugu vrstu podataka čine realni primeri: tekst knjige od 3226571 karaktera i CIM/XML fajl sa 2GB podataka.

U tabeli 1, prikazano je vreme izvršavanja algoritama nad veštački generisanim primerima, kao i pri različitim dužinama obrasca. Tekstualni fajl nad kojim su izvršena testiranje je bio veličine 274 MB (tj. sadrži 280302624 karaktera).

Tabela 1. *Vremena izvršavanja algoritama nad veštački generisanim primerima*

Obrazac	*m*	BF [s]	KMP [s]	RK [s]	BM [s]	Broj pogodaka
aaaaa aaaab	10	15.75	4.80	6.70	10.6	6673872
aaaaa aaaac	10	15.78	5.55	6.47	9.89	0
aaaaa... ...aaaab	42	37.94	4.38	8.14	9.12	6673872
aaaaa... ...aaaac	42	38.12	7.40	7.57	1.84	0

Kao što se može videti iz ovog primera, BF algoritam ima duže vreme izvršavanja kako se povećava dužina obrasca i gotovo da nije bitno da li ima ili nema pogodaka.

Što se tiče KMP algoritma, najveća razlika se vidi kada ima i kada nema podudaranja. Razlog zašto je algoritam sporiji kad nema podudaranja je taj što algoritam koristi pomoćnu tabelu koja govori sa kojim karakterom obrasca bi trebalo izvršiti poređenje, i tako se pojavljuju dodatna poređenja koja povećavaju vreme izvršavanja.

Na ovom primeru, RK algoritam ima približno isto vreme izvršavanja kada ima i kada nema podudaranja. Takođe, ima malo duže vreme izvršavanja za duže obrasce.

Sa druge strane na BM algoritam znatno utiče veličina obrasca jer algoritam ima mogućnost da u određenim slučajevima preskoči deo teksta za celu dužinu obrasca, što se i desilo u ovom primeru, i iz toga se vidi koliko BM algoritam može biti brži od ostalih u takvim slučajevima.

U drugom testu su algoritmi testirani nad „pravim“ tekstom. U pitanju je knjiga Lava Tolstoja, Rat i mir. Rezultati drugog testa su prikazani u tabeli broj 2.

Tabela 2. *Vremena izvršavanja algoritama nad knjigom Rat i mir*

Obrazac	*m*	BF [s]	KMP [s]	RK [s]	BM [s]	Broj pogodaka
a	1	0.08	0.08	0.10	0.16	195215
been	4	0.09	0.07	0.09	0.07	1476
suddenly	8	0.10	0.08	0.09	0.05	432
illustrious	11	0.09	0.11	0.11	0.05	2

Iz predhodnog primera, pri pretraživanju stvarnog teksta, možemo videti da su rezultati približno jednaki za različite dužine obrazaca. Ukoliko je obrazac dužine jedan karakter, tada je BF metod najbrži, a BM najsporiji, iz razloga što je preskakanje teksta minimalno, što je glavna prednost BM algoritma, i dodatne provere za potencijalno preskakanje su usporile algoritam u odnosu na BF.

Kod najdužeg obrasca BM algoritam ima najbolje vreme izvršavanja, dok RK algoritam ima najsporije vreme izvršavanja.

U trećem testu su algoritmi testirani nad CIM/XML fajlom veličine 2GB. CIM/XML fajl služi za skladištenje, obradu i razmenu podataka. Podaci se sastoje od naziva elementa, koji se nalaze u tagovima, i sadržaja koji daje određenu vrednost elementu i nalazi se između tagova. Elementi, takođe, mogu imati i atribute, koji ih dodatno definišu. Rezultati pretrage nad CIM/XML fajlom su prikazani u tabeli 3.

Tabela 3. *Vreme izvršavanja algoritama nad cim/xml fajlom*

Obrazac	*m*	BF [s]	KMP [s]	RK [s]	BM [s]	Broj pogodaka
agms	4	14.5	19.1	25.5	16.3	19836585
altitude	8	13.4	19.3	24.9	9.3	205014
_100O4L7KZ5X	12	12.6	19.9	25.4	7.3	9
PositionPoint.x Position	23	13.7	21.7	27.1	11.6	4400534

U primeru sa pretragom CIM/XML fajla, za kraće obrasce, dužine do 4 karaktera, najbolje vreme ima *brute force* algoritam, dok najgore vreme ima Rabin-Karp algoritam, iz razloga što je fajl prevelik, i ima dosta pojmova, što znači da algoritam mora da za svaki računa heš funkciju što usporava vreme izvršavanja algoritma.

Prilikom većeg broja karaktera obrasca, Boyer-Moore daje znatno bolje vreme izvršavanja, naročito u slučaju malog broja podudaranja, iz razloga manjeg broja provera u samom algoritmu.

Knuth-Morris-Pratt i Rabin-Karp daju približno isto vreme izvršavanja i kod pretrage realnog teksta, a i kod pretrage CIM/XML fajla.

4. ZAKLJUČAK

U ovom radu su predstavljeni algoritmi *Brute force*, Knuth-Morris-Pratt, Rabin-Karp i Boyer-Moore za pretragu podudaranja obrazaca. Opisani su i varijacije Boyer-Moore algoritma, Boyer-Moore-Horspool i Boyer-Moore-Horspool-Sunday algoritmi. Za svaki od algoritama je opisan način rada i date su složenosti algoritama za najbolje i najgore slučajeve.

Algoritmi su implementirani u C# programskom jezijku i izvršeno je poređenje performansi nad testnim skupovima podataka. Prvo je testirano nad veštački generisanim podacima, dok je sa druge strane testirano nad realnim primerima u vidu pretrage teksta iz knjige i pretrage CIM/XML fajla.

U osnovi, svaki od ovih algoritama donosi dobre rezultate ukoliko je tekst kratak, ali ukoliko se testira nad velikom količinom podataka, i velikim obrascem, vidi se najveća razlika algoritama. Dolazi se do zaključka da ukoliko je obrazac kratak, oko 4 karaktera dužine, najbolje vreme izvršavanja daje Brute force metoda, dok ukoliko je obrazac preko 8 karaktera dužine, najbolje vreme izvršavanja ima Boyer-Moore algoritam.

Najsporije vreme izvršavanja u skoro svim segmentima ima Rabin-Karp algoritam. Knuth-Morris-Pratt algoritam ima približno isto vreme u praksi za različite dužine obrasca.

5. LITERATURA

[1] Herbert S. Wilf, "Algorithms and Complexity", Summer 1994.

[2] C. Charras, T. Lecroq, „Handbook of Exact String-Matching Algorithms", 2004.

[3] http://staff.ustc.edu.cn/~csli/graduate/algorithms/book6/chap34.htm (pristupljeno u septembru 2019.)

[4] D. Gusfield, "Algorithms on Strings, Trees and Sequences. *Computer science and computational biology"*, 1997.

[5] https://www.geeksforgeeks.org/boyer-moore-algorithm-for-pattern-searching/ (pristupljeno u septembru 2019.)

[6] Ramshankar Choudhary, „Variation of Boyer-Moore String Matching Algorithm: A Comparative Analysis", February 2012.

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, „Introduction to algorithms", 2001.

Kratka biografija:

Branislav Trkulja rođen je u Novom Sadu 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Primenjeno softversko inženjerstvo odbranio je 2019.god. kontakt: branislavtrkulja@hotmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/07BE49Corilic

GENERISANJE AKREDITACIONE DOKUMENTACIJE IZ ONTOLOGIJE STUDIJSKIH PROGRAMA

GENERATION OF ACCREDITATION DOCUMENTATION FROM ONTOLOGY OF STUDY PROGRAMS

Branko Ćorilić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je opisana prototipska implementaciju aplikacije za generisanje dokumentacije za akreditaciju iz ontologije studijskih programa. Za opis studijskih programa korišćen je model meta podataka MLO ECTS IP/CC (ECTS Information Package/Course Catalogue MLO Application Profile). Kao mustra za generisanje dokumentacije korišćen je Word dokument kao i OpenXML (zipovan, XML baziran format razvijen od strane Microsoft-a).*

Ključne reči: *Akreditaciona dokumentacija, MLO ECTS IP/CC, ontologije, studijski program*

Abstract – *This paper describes a prototype implementation of an application for generating accreditation documentation from an ontology study program. The MLO ECTS IP / CC meta data model (ECTS Information Package / Course Catalog MLO Application Profile) was used to describe the study programs. As a template for documentation generation a Word document was used and OpenXML (zipped, XML based format developed by Microsoft)*

Keywords: *Accreditation documentation, MLO ECTS IP/CC, ontologies, study program*

1. UVOD

Akreditacija visokoškolskih ustanova i studijskih programa visokog obrazovanja predstavlja proces u kome se formiraju uporedni međunarodni pokazatelji kojima se afirmiše kvalitetno obrazovanje. Po Bolonjskoj deklaraciji i Lisabonskoj konvenciji ona predstavlja jedan od preduslova za lakšu razmenu studenata i zapošljavanje diplomiranih stručnjaka. Sam proces izrade dokumentacije je prilično složen i podrazumeva zadovoljavanje određenih akreditacionih parametara potrebnih za akreditovanje studijskih programa i same visokoškolske ustanove. Neki od problema sa kojima se obrazovne institucije susreću pri procesu pripreme dokumentacije za akreditaciju su:

• preobilna dokumentacija (za različite standarde zahtevana je ista dokumentacija)

• promena parametara (promene u studijskom programu, promene nastavnog kadra) koje mogu da utiču i na više studijskih programa

• kratak vremenski rok

Jedan od načina kako bi se ovi problemi mogli prevazići jeste predstavljanje dokumenata potrebnih za akreditaciju tehnologijama Semantičkog Veba (ontologijama) koje bi omogućile njihovo lakšu organizaciju, upravljanje po sadržaju, semantici i domenu, automatsko generisanje određenog dela dokumentacije.

Glavni motiv ovog rada jeste potreba da se proces kreiranja dokumentacije učini efikasnijim, bržim i manje sklonim greškama. Jedan od načina da se to učini jeste automatizacija procesa izrade dokumentacije gde bi se dokumentacija generisala programski, a izvor podataka bi bile popunjene ontologije koje predstavljaju akreditacionu dokumentaciju.

2. TEORIJSKE OSNOVE

2.1. Ontologije

Ontologije su deo W3C grupe standarda za Semantički Veb. Predstavljaju formalnu deljenu specifikaciju konceptualizacije [1]. To je formalni opis znanja kao skupa pojmova unutar nekog domena i odnosa koji postoje među njima.

2.2 Office Open XML

Office Open XML, takođe poznat kao OpenXML ili OOXML, je XML zasnovan format za Office dokument, uključujući dokumente za obradu teksta, proračunske tabele, prezentacije, kao i grafikone, dijagrame, oblike i drugi grafički materijal. Specifikaciju je razvio Microsoft, a *ECMA International* je usvojila kao ECMA-376 2006. Druga verzija je objavljena u decembru 2008., a treća verzija standarda objavljena u junu 2011. Specifikaciju su usvojili ISO i IEC kao ISO / IEC 29500.

2.3 Bolonjski proces i proces akreditacije

Bolonjski proces je masivni, višeslojni projekat sa ciljem formiranja evropske oblasti visokog obrazovanja (*European Higher Education Area*, EHEA) [2]. Bolonjskim procesom se želi postići veća koherentnost u visokoškolskim sistemima širom Evrope.
Ovim procesom je uspostavljen evropski prostor visokog obrazovanja kako bi se olakšala mobilnost studenata i osoblja, učinilo visoko obrazovanje inkluzivnijim i dostupnijim, a visoko obrazovanje u Evropi učinilo atraktivnijim i konkurentnijim širom sveta.

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

2.4 MLO-AD

Metapodaci za obrazovna prilike (*Metadata for Learning Opportunities -Advertising*, MLO-AD) (CEN WS-LT, 2012) su model za predstavljanje meta podataka dovoljnih za opis i oglašavanje obrazovnih prilika. Model je dizajniran tako da bude prilagođen upotrebi semantičkih tehnologija i web arhitektura i da podrži različite mehanizme za razmenu i agregaciju informacija o obrazovnim prilikama. [3] Ovaj model je usvojen kao evropski standard CWA 15903:2008 (CEN WS-LT, 2008)

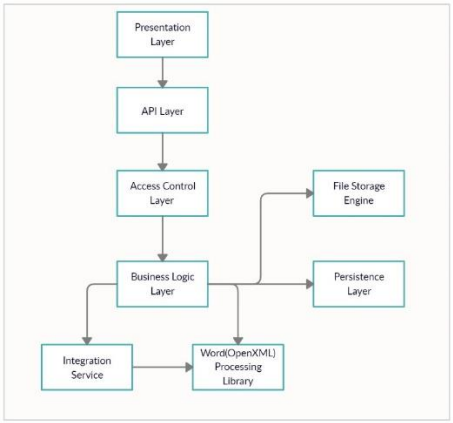
2.5 MLO ECTS IP/CC

ECTS informacioni paket/ katalog kurseva za MLO aplikacione profile (ECTS IP/CC) predstavlja proširenje MLO-AD definisano od strane evropskog komiteta za standardizaciju [4] kojim je definisan standard za predstavljanje kredita u skladu sa Evropskim sistemom prenosa i akumulacije kredita. Drugim rečima ovaj model predstavlja obrazac za opisivanje studija, nastavnih jedinica i visokoškolskih ustanova koje ih nude.

3. ARHITEKTURA SISTEMA

Arhitektura rešenja sastoji se iz sledećih sedam komponenti:

- Presentation Layer
- API Layer
- Access Control Layer
- Business Logic Layer
- Integration Service
- Persistence Layer
- Word (OpenXML) Processing Library
- File Storage Engine



Slika 2.1 - *Arhitektura prototipske implementacije aplikacije*

3.1. Presentation Layer

Presentation Layer predstavlja korisnički interfejs koji omogućava korisniku upravljanje aplikacijom. Pomoću ovog sloja korisnik može da:

- unosi mustre u obliku Word dokumenta
- ima pregled unesenih mustri
- administrira mustre
- pretražuje mustre
- unosi/ menja markere koji služe kao *placeholderi* unutar mustre koji će se zameniti podacima iz ontologije prilikom generisanja dokumentacije
- Generiše dokumentaciju tako što će ručno popuniti unete mustre
- Generiše dokumentaciju automatski unosom ontologije studijskih programa

U ovoj aplikaciji korisnički interfejs predstavlja posebnu web aplikaciju koja je strukturno odvojena od ostatka aplikacije i komunicira sa Business Logic Layer-om REST pozivima koristeći HTTPS (*Secure Hypertext Transfer Protocol*).

3.2. Access Control Layer

Access Control predstavlja posebnu aplikaciju koja je zadužena za autentifikaciju i autorizaciju klijenata. Implementirana je kao posebni mikro servis u jeziku .Net Core. Podatke o klijentima (*Tenant*) čuva u odvojenoj MySQL bazi (tenantdb) kojoj samo ona ima pristup.
Komunikacija između Access Control aplikacije i Business Logic Layer-a omogućen je putem AccessControl klijentske biblioteke. Biblioteka je napisana tako da u potpunosti podržava DI prirodu .Net Core web aplikacija.

3.3. API Layer

API Layer predstavlja komponentu koja za cilj ima da omogući interakciju između korisničkog interfejsa i biznis logike aplikacije. Ona omogućava kreiranje RESTful servisa. Sastoji se iz vise *controller* klasa u kojima su definisani *endpoints* preko kojih je moguća komunikacija između korisničkog interfejsa i biznis logike. Implementirana je po DI parternu.

3.4. Business Logic Layer

Business Logic Layer je centralna komponenta koja povezuje sve druge komponente u sistemu. Njena uloga je da obavlja „biznis logiku“, tj. određuje set pravila koji upravljaju razmenom informacija između baze podataka (*Persistance Layer*) i korisničkog interfejsa (*Presentation Layer*), određuje logiku kako podaci su kreirani, čuvani ili promenjeni.
U ovoj aplikaciji biznis logika je implementirana u obliku servisa od kojih svaki od njih ima svoj interfejs što omogućava da ovakva arhitektura bude u potpunosti u skladu sa DI paternom koji se koristi kroz aplikaciju.

3.5. Persistence Layer

Persistence Layer je komponenta koja služi za skladištenje podataka u aplikaciji (markera, podacima o mustrama). Ovoj komponenti podaci se prosleđuju preko Business Logic Layer-a i ona omogućava skladištenje podataka u MySql bazu podataka.
Cilj ove komponente je da omogući nezavisnu komunikacijo aplikacije sa bazom podataka od biznis logike aplikacije.
Prilikom implementacije ovog sloja nije korišćen ni jedan gotov *Framework* (npr. Entity Framework) za manipulaciju objektima sa MySql bazom već je dato svoje rešenje u obliku ručno pisane Entity Manager biblioteke.

3.6. Word (OpenXML) Processing Library

Word (OpenXML) Processing Library-a je biblioteka koja manipuliše Word dokumentima koristeći OpenXML. U ovoj aplikaciji uloga ove biblioteke je višestruka:

- Služi za generisanje grafičke reprezentacije mustre koje je korisnik uneo preko korisničkog interfejsa uplodovanjem Word dokumenta; Prilikom uplodovanja Word dokument koji je u osnovi XML mapira se u odgovarajući HTML
- Služi za zamenu markera unutar mustre sa odgovarajućim podacima dobijenih od integracionog sloja

Omogućava automatsko ekstrahovanje markera iz Word dokumenta koji predstavlja mustru

3.7. File Storage Engine

File Storage Engine (FSE) predstavlja posebnu aplikaciju koja služi upravljanje i čuvanje dokumenta. Osnovna ideja File Storage Engine-a je da se dokumenti čuvaju na hard disku kompjutera unutar direktorijuma i pod direktorijuma a meta podaci o dokumentima unutar relacione MySQL baze.

4. VERIFIKACIJA

U ovom poglavlju opisana je prototipska implementacija aplikacije za generisanje dokumentacije za akreditaciju iz ontologije studijskih programa. Ova implementacija je opisana kroz praćenje procesa neophodnih za uspešan završetak zadatka, tako da je ovo poglavlje podeljeno na 3 dela:

- Administracija markera
- Administracija mustri
- Generisanje dokumentacije iz ontologije

4.1. Administracija markera

U ovoj aplikaciji markeri predstavljaju označavajuće reči koji se umeću unutar Word dokumenta. Njihova uloga je dvojaka. Prilikom generisanja korisničkog interfejsa oni se generišu u polja za unos podataka a prilikom generisanja dokumentacije oni služe kao objekti na koje se mapiraju odgovarajući podaci iz ontologije.
Postoje 2 tipa markera, prosti koji predstavljaju 1 objekat i složeni koji služe za predstavljanje lista objekata.

Da bi prosti marker bio dobro formiran unutar Word dokumenta on mora da počinje sa {{ i završava se sa }} zagradama.
Složeni markeri predstavljaju listu objekata koja treba da se zameni prilikom generisanja dokumentacije. U osnovi ovaj marker je zamišljen kao trodimenzionalni niz veličine [1][n][m] gde prva dimenzija predstavlja identifikator tipa objekta, druga prestavlja atribute objekta a treća listu vrednosti atributa. U Word dokumentu ovi markeri mogu imati primenu prilikom generisanja složenih struktura kao što su tabele ili liste gde je za generisanje, kao izvor podataka, potrebno proslediti niz složenih objekata gde svaki objekat predstavlja jedan red. Slika 2.2 prikazuje primer korišćenja složenog markera.

Литература				
Р.бр.	Аутор	Назив	Издавач	Година
${literatura}{r_br}	${literatura}{autor}	${literatura}{naziv}	${literatura}{izdavac}	${literatura}{godina}

Slika 2.2 – *Prikaz korišćenja složenog markera*

4.1.1. Ručni unos markera

Markeri mogu ručno da se kreiraju i ažuriraju preko forme za ručno ažuriranje. Razlog zbog čega je data mogućnost da se ručno unose markeri je zbog toga što se ti markeri pretvaraju prilikom generisanja UI prikaza obrasca u polja za unos podataka koji imaju *placeholder* atribut koji je moguće koristiti za opis polja.

4.1.2 Automatski unos markera

Pored ručnog unosa moguće je da se markeri kreiraju automatski prilikom procesa kreiranja mustre za generisanje dokumentacije.Markeri se unutar Word dokumenta, koji predstavlja novu mustru, označavaju po principu {{ime_markera}}. Kada se Word dokument unese putem forme za unos Word (OpenXML) Processing Library očitava dokument i pretražuje sve markere unutar njega. Zatim proverava se da li su ti markeri već sačuvani u MySQL bazu. Ako nisu onda se kreira novi marker gde je ime markera vrednost između {{ i }} zagrade.

4.2. Administracija mustri

Pre početka generisanja same dokumentacije za akreditaciju iz ontologije studijskih programa pored kreiranja markera potrebno je i uneti Word dokumenta sa markerima koji će predstavljati mustru za generisanje.

4.2.1 Unos mustre za generisanje dokumentacije

Prvi korak u procesu kreiranja nove mustre za generisanje dokumentacije jeste da korisnik definiše Word dokument koji prestavljati mustru za generisanje. U tom Word dokumentu potrebno je da sam ručno doda markere koje treba da zamene podaci iz ontologije pri automatskom generisanju. (pored ontologije izvor podataka može biti i Excel dokument). Pri korišćenju markera treba da se pazi da razmaci između {{ i }} se ne ignorišu i da su markeri *case sensitive*. Slika 2.3 prikazuje primer Word dokumenta koji je služio kao mustra za generisanje dokumentacije.

Slika 2.3 – *Prikaz Word dokumenta sa popunjenim markerima za mustru*

Nova mustra se unosi preko forme za ažuriranje mustri. Uplodovani dokument se prvo šalje File Storage Engine-a koji ga sačuva i vraća ID originalnog sačuvanog dokumenta.
Zatim dokument se skenira pomoću Word (OpenXML) Processing biblioteke koja dokument parsira kao jedan XML objekat (pomoću OpenXML). Iz tog objekta izvlače se svi markeri, proveravaju da li su sačuvani u bazi podataka, kreiraju novi koji već nisu sačuvani. Zatim se

XML reprezentacija Word dokumenta konvertuje u HTML dokument a pri tom se pazi da vizualni prikaz Word dokumenta ostane isti i da atributi i elementi iz XML objekta mapiraju na odgovarajuće elemente i atribute u HTML-u.
Sledeći korak je zamena markera odgovarajućim poljima za unos. Name atribut markera mapira se na *ng-model* atribut a HTMLPlaceholderValue na atribut *placeholder*. Ovakvo mapiranje omogućava da polja za unos kreirana od istih markera uvek bude isto popunjena.
Kao Http Response vraća se objekat koji sadrži ID originalnog sačuvanog dokumenta i HTML koji vizualno predstavlja unetu mustru u aplikaciji.
Na korisničkom interfejsu HTML koji je izgenerisan se prikazuje korisniku. To je moguće zbog AngularJS $sce (*Strict Contextual Escaping*) servisa i njegove metode *trustAsHtml(vrednost)* koja omogućava generisanje HTML-a.
Da bi mustra mogla da se koristi dalje u procesu generisanja dokumentacije potrebni je uneti njen naziv i na kraju pritisnuti dugme sačuvaj.

4.3. Generisanje dokumentacije iz ontologije

Generisanje dokumentacije se vrši tako što se prvo sa forme obrasci izabere odgovarajuća mustra na osnovu koje će se vršiti generisanje. Prilikom odabira mustre otvara se forma za generisanje. Na toj formi postoje dva načina da se generiše dokumentacija:

1. Ručno popunjavanjem polja za unos na mustri
2. Automatski tako što će se uneti popunjena ontologija kao izvor podataka

4.3.1 Integracioni sloj

U slučaju da korisnik odluči da generiše dokumentaciju automatski potrebno je da uploduje odgovarajuću ontologiju iz koje će se izvući podaci potrebni za generisanje. U ovom zadatku za potrebe generisanja dela akreditacione dokumentacije korišćena je ECTS IP/CC ontologija koja predstavlja studijske programe.
Ontologija je popunjena konkretnim podacima i oni su u ontologiji predstavljeni kao *Individuals.* Za vađenje podataka iz ontologije korišćena je gotova *open source* biblioteka OwlDotNetApi koja predstavlja RDF *driven* parser za C# .NET i u potpunosti je u skladu sa W3C OWL sintaksom.
Mapiranje izvučenih podataka iz ontologije na strukturu koja odgovara markerima u mustri je u ovoj prototipskoj aplikaciji obavljeno ručno, međutim autor predlaže da se u budućnosti aplikacija proširi i da se omogući korisniku mogućnost da preko posebne forme mapira objekte iz učitane ontologije na odgovarajuće markere i na taj način podaci mapiraju automatski na potrebnu strukturu.

4.3.2 Manipulacija originalnog dokumenta i generisanje

Kada se podaci izvuku iz ontologije i mapiraju na strukturu koja je potrebna za generisanje pomoću File Storage Engine-a se vraća originalni dokument sa hard diska. Pravi se kopija dokumenta i ona se otvara uz pomoć Word (OpenXML) Processing Library biblioteke. Zatim se uz pomoć metode:

ReplaceAllPlaceholdersAndMergeDocumentsInOne

pronađu svi markeri u dokumentu i zamene sa podacima iz ontologije. Ako su podaci iz ontologije struktuirani tako da prestavljaju listu objekata (npr. listu studijskih programa) generiše se onoliko Word dokumenata koliko ima objekata u listi. Na kraju se svi ti Word dokumenti spoje u jedan i kao takav vrate korisniku koji taj novo kreirani Word dokument skine na svoj kompjuter. Time je gotov proces automatskog generisanja opisan u ovom radu.

5. ZAKLJUČAK

U okviru ovog rada predstavljena je prototipska implementacija aplikacije koja olakšava kreiranje dela dokumentacije za akreditaciju. Ova aplikacija automatizuje proces generisanja dokumentacije potrebne za akreditaciju studijskih programa iz ontologije koja predstavlja podatke o studijskim programu.
Za opis studijskih programa korišćen je model meta podataka MLO ECTS IP/CC (*ECTS Information Package/ Course Catalogue MLO Application Profile*). Kao mustra za generisanje korišćen je Word dokument u kome se nalaze označavajuće reči (markeri) koje označavaju mesto gde treba da se mapiraju podaci. Open XML jezik korišćen je za manipulaciju originalnog Word dokumenta, kao i mapiranje podataka neophodnih za generisanje iz ontologije na mustru.
Ovim rešenjem postignuta je osnova za brzo automatsko generisanje dokumentacije za akreditaciju.
Mapiranje izvučenih podataka iz ontologije na strukturu koja odgovara markerima u mustri je u ovoj prototipskoj aplikaciji obavljeno ručno.
Budući rad na ovoj aplikaciji bi trebao da bude usmeren ka tome da se omogući generisanje i drugih delova dokumentacije potrebnih za akreditaciju. Jedan od načina kako bi se to moglo postići je omogućavanje korisniku da sam dinamički, preko aplikacije, mapira unesene ontologije sa markerima u unetim mustrama. Na taj način bi se omogućilo generisanje bilo koje dokumentacije iz odgovarajuće ontologije.

6. LITERATURA

[1] Antoniou, Grigoris, and Frank Van Harmelen. *A semantic web primer*. MIT press, 2004.

[2] Terry, Laurel S. "The Bologna Process and its Impact in Europe: it's so much more than degree changes." Vand. J. Transnat'l L. 41 (2008): 107.

[3] Milan Segedinac, *Razvoj proširive softverske platforme za upravljanje kurikulumom u internacionalizovanom visokom obrazovanju*, doktorska disertacija, 2014

[4]WS-LT, CEN. "ECTS Information Package." *Course Catalogue MLO Application Profile* (2010).

Kratka biografija:

Branko Ćorilić rođen je u Novom Sadu 1989. god. Master rad na Fakultetu tehničkih nauka iz oblasti Primenjene računarske nauke i informatika odbranio je 2019.god.
kontakt: corilic@gmail.com

U realizaciji Zbornika radova Fakulteta tehničkih nauka u toku 2019. godine učestvovali su sledeći recenzenti:

Aco Antić
Aleksandar Erdeljan
Aleksandar Ristić
Bato Kamberović
Biljana Njegovan
Bogdan Kuzmanović
Bojan Batinić
Bojan Lalić
Bojan Tepavčević
Bojana Beronja
Branislav Atlagić
Branislav Nerandžić
Branislava Novaković
Branka Nakomčić
Branko Milosavljević
Branko Škorić
Damir Đaković
Danijela Lalić
Darko Čapko
Darko Marčetić
Darko Reba
Dejan Ubavin
Dejana Nedučin
Dragan Ivanović
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
Ivan Beker
Ivana Katić
Ivana Kovačić
Ivana Miškeljin
Jasmina Dražić
Jelena Atanacković Jeličić
Jelena Borocki
Jelena Kiurski
Jelena Radonić
Jovan Petrović
Jovanka Pantović
Ksenija Hiel
Laslo Nađ
Lazar Kovačević
Leposava Grubić Nešić
Livija Cvetičanin
Ljiljana Vukajlov
Ljiljana Cvetković
Ljubica Duđak
Maja Turk Sekulić
Marko Todorov
Marko Vekić
Maša Bukurov
Matija Stipić
Milan Rapajić
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
Mirjana Radeka
Mirko Borisov
Miro Govedarica
Miroslav Hajduković
Miroslav Popović
Mitar Jocanović
Mladen Kovačević
Mladen Radišić
Nemanja Stanisavljević
Nemanja Sremčev
Nikola Đurić
Nikola Jorgovanović
Nikola Radaković
Ninoslav Zuber
Ognjen Lužanin
Pavel Kovač
Peđa Atanasković
Petar Malešev
Predrag Šiđanin
Radivoje Dinulović
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 Nikoličić
Tanja Kočetov
Tatjana Lončar - Turukalo
Uroš Nedeljković
Valentina Basarić
Velimir Čongradec
Veran Vasić
Veselin Perović
Vladimir Katić
Vladimir Strezoski
Vlado Delić
Vlastimir Radonjanin
Vuk Bogdanović
Zdravko Tešić
Zoran Anišić
Zoran Brujić
Zoran Jeličić
Zoran Mitrović
Zoran Papić
Željen Trpovski
Željko Jakšić