УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

ЗБОРНИК РАДОВА
ФАКУЛТЕТА ТЕХНИЧКИХ НАУКА

Едиција: Техничке науке – зборници
Година: XL
Број: 7/2025

Нови Сад

Едиција: „Техничке науке – Зборници“
Година: XL Свеска: 7
Издавач: Факултет техничких наука Нови Сад

Главни и одговорни уредник: проф. др Борис Думнић, декан Факултета техничких наука у Новом Саду

Савет за библиотечку и издавачку делатност ФТН, проф. др Селена Самарџић Цвијановић, председник.

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

CIP-Каталогизација у публикацији
Библиотека Матице српске, Нови Сад

378.9(497.113)(082)
62

ЗБОРНИК радова Факултета техничких наука / главни и одговорни уредник
Борис Думнић. – 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 седма 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».

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

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

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

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

У свесци са редним бројем 7, објављени су радови из области електротехнике и рачунарства.

Континуираним радом и унапређењем квалитета часописа, план је да часопис постане препознатљив међу ауторима, чиме ће значајно допринети да се оствари мото Факултета техничких наука:

„Високо место у друштву најбољих“

Уредништво

Sadržaj

Elektrotehnika i računarstvo

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.438
DOI: https://doi.org/10.24867/31BE01Milovanovic

PLATFORMA ZA STRIMOVANJE VIDEO SADRŽAJA ZASNOVANA NA MIKROSERVISNOJ I SERVERLES ARHITEKTURI

VIDEO STREAMING PLATFORM BASED ON MICROSERVICE AND *SERVERLESS* ARCHITECTURE

Milovan Milovanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO
Kratak sadržaj – *Ovaj rad se bavi unapređenjem mikroservisne platforme za strimovanje video sadržaja implementirane u ranijem diplomskom radu. U radu je opisana primena serverles (engl. serverless) arhitekture i AWS servisa kao što su Lambda, S3 i CloudFront sa ciljem da platforma postane globalno dostupna i skalabilna. U radu je takođe dat opis primene AWS CDK i Serverless Framework alata za definisanje infrastrukture putem koda.*
Ključne reči: *serverles, strimovanje videa, AWS, Lambda, IaC, CDK*
Abstract – *This paper focuses on improving the microservice platform for video streaming developed in the bachelor thesis. It describes the application of serverless architecture and AWS services such as Lambda, S3, and CloudFront, with the aim of making the platform globally accessible and scalable. The paper also provides an overview of the use of AWS CDK and Serverless Framework tools for defining infrastructure as code.*
Keywords: *serverless, video streaming, AWS, Lambda, IaC, CDK*

1. UVOD

Sa porastom popularnosti video-sadržaja na internetu, javlja se potreba za platformama koje mogu efikasno upravljati i isporučivati multimedijalni sadržaj korisnicima širom sveta. Striming servisi kao što su *YouTube* i *Netflix* postali su integralni deo svakodnevnog života, postavljajući nove standarde za skalabilnost, pouzdanost i brzinu isporuke video-sadržaja. Međutim, razvoj ovakvih platformi nije trivijalan i zahteva rešavanje složenih problema poput upravljanja velikim količinama podataka, visoke dostupnosti i optimizacije troškova infrastrukture.

Cilj ovog rada jeste da pruži jedno moguće rešenje koje će odgovoriti na navedene probleme. Polazna tačka rešenja je diplomski rad pod naslovom „Platforma za gledanje i deljenje video-sadržaja bazirana na mikroservisnoj arhitekturi“, u okviru kojeg je razvijena osnovna aplikacija za upravljanje video-sadržajima. Ta aplikacija je, u svojoj inicijalnoj verziji, funkcionisala isključivo u lokalnom okruženju.

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

U ovom radu, inicijalno rešenje je prošireno i nadograđeno kroz primenu *AWS* (*Amazon Web Services*) servisa tzv. računarstva u oblaku (engl. *cloud computing*) i serverles (engl. *serverless* – bez servera) arhitekture, odnosno arhitekture „bez servera“, što omogućava postavljanje aplikacije u oblak, sa potpunom podrškom za skalabilnost, automatsko upravljanje resursima i troškovnu efikasnost. Kroz integraciju servisa kao što su *AWS Lambda* za izvršavanje bez servera, *S3* za skladištenje, *RDS* za bazu podataka i *CloudFront* za isporuku sadržaja, platforma je transformisana u globalno dostupan sistem koji može da odgovori na izazove modernih video-servisa.

U okviru rešenja koristiće se i drugi *AWS* servisi kao što su *API Gateway* za upravljanje *API* zahtevima (*HTTP/REST API* i *WebSocket API*), *DynamoDB* za skladištenje podataka koji zahtevaju visoke performanse, *VPC* za mrežnu i bezbednosnu izolaciju resursa, i *Secrets Manager* za sigurno upravljanje osetljivim podacima. Pored toga, primenjuju se *AWS CDK* za infrastrukturu kao kod (engl. *infrastructure as code*; *IaC*), odnosno definisanje infrastrukture u kodu, kao i *Serverless Framework* za orkestraciju i postavljanje serverles funkcija u oblak.

2. PREGLED FUNKCIONALNOSTI SISTEMA

U ovom poglavlju dat je kratak pregled funkcionalnosti platforme za gledanje i deljenje video-sadržaja implementiranih u diplomskom radu [1], koji je polazna tačka ovog rada. U nastavku su opisani tipovi korisnika u sistemu, kao i same funkcionalnosti.

2.1. Tipovi korisnika

Sistem poznaje četiri tipa korisnika [1]: neregistrovan korisnik, registrovan korisnik, administrator i tehnička podrška.

2.2. Funkcionalnosti

U daljem tekstu su opisane dostupne funkcionalnosti po tipu korisnika u sistemu [1].

Neregistrovan korisnik: registracija na sistem, prijava na sistem, pretraga videa u sistemu i gledanje odn. strimovanje videa [1].

Registrovan korisnik: pretraga videa u sistemu, strimovanje videa, aploadovanje videa na sistem, brisanje sopstvenih videa sa sistema, ocenjivanje videa, pregled prosečne

ocene videa, pregled komentara na videu, postavljanje i brisanje sopstvenih komentara na videu, prijava komentara, prijava videa, pregled i izmena profila, komunikacija sa tehničkom podrškom i odjava sa sistema [1].

Administrator: sve funkcionalnosti registrovanog korisnika osim aploadovanja videa na sistem i komunikacije sa tehničkom podrškom, brisanje videa, brisanje komentara, uvid u prijavljene videe i komentare, kao i brisanje sadržaj uz blokiranje korisnika [1].

Tehnička podrška: prijava na sistem, pregled i izmena profila i komunikacija sa registrovanim korisnicima [1].

3. KORIŠĆENE TEHNOLOGIJE I KONCEPTI

Za proširenje platforme iz [1], kojim se ovaj rad bavi, relevantni su određeni koncepti i tehnologije koje se primenjuju kako bi platforma postala globalno dostupna, skalabilna i troškovno efikasna. Opis tih koncepata i tehnologija dat je u nastavku.

3.1. Računarstvo u oblaku

Računarstvo u oblaku (engl. *cloud computing*) predstavlja model računarskih usluga koji se zasniva na konceptu deljenja računarskih resursa, softvera i informacija na zahtev putem interneta. Kompanije ili pojedinci plaćaju kako bi pristupili virtuelnom fondu deljenih resursa, koji obuhvataju računarsku snagu, skladištenje i mrežne usluge. Ovi resursi su smešteni na udaljenim serverima koje poseduju i održavaju pružaoci *cloud* usluga [2].

3.2. Serverles arhitektura i *AWS Lambda*

Serverles arhitektura (u bukvalnom prevodu „arhitektura bez servera“) pristup je dizajnu softvera gde programeri mogu da razvijaju i upravljaju aplikacijama bez upravljanja osnovnom infrastrukturom. Uprkos možda zbunjujućem nazivu, serverles aplikacije se i dalje izvršavaju na serverima, ali je provajder odn. pružalac *cloud* usluga odgovoran za održavanje, obezbeđivanje, upravljanje i skaliranje celokupne infrastrukture u oblaku [3].

AWS Lambda je usluga serverles računarstva koju pruža *Amazon Web Services* (*AWS*). Korisnici *AWS Lambda* servisa kreiraju funkcije – samostalne aplikacije napisane na jednom od podržanih jezika i okruženja, i otpremaju ih u *AWS Lambda*, koji te funkcije izvršava na efikasan i fleksibilan način [4].

Glavne prednosti *AWS Lambda* servisa u odnosu na održavanje sopstvenih servera u oblaku su: plaćanje po korišćenju, infrastruktura kojom upravlja provajder, automatsko skaliranje (što znači da nema nivoa skaliranja i drugih podešavanja o kojima treba brinuti – funkcije su dostupne kad god se opterećenje poveća ili smanji), kao i laka integracija sa drugim *AWS* servisima [4].

3.3. *Amazon API Gateway*

Amazon API Gateway je servis koji olakšava programerima da kreiraju, objavljuju, održavaju, nadgledaju i obezbeđuju *API*-je u *AWS* oblaku. Koristeći *API Gateway*, mogu se kreirati *HTTP/REST API*-ji, kao i *WebSocket API*-ji, koji omogućavaju potpuno dvostranu komunikaciju u realnom vremenu (engl. *real-time*) između servera i klijenata [5].

API Gateway može da se kombinuje sa *AWS Lambda* servisom kako bi se realizovala serverles arhitektura. Kada se kombinuju, *API Gateway* postaje ulazna tačka za *Lambda* funkcije, omogućujući krajnjim korisnicima (ili aplikacijama i servisima) da šalju *HTTP(S)* ili *WebSocket* zahteve ka *Lambda* funkcijama.

3.4. *Amazon S3*

Amazon S3 (*Amazon Simple Storage Service*) je servis za skladištenje koji čuva podatke kao objekte unutar baketa (engl. *buckets*). Objekat predstavlja datoteku zajedno sa svim metapodacima koji opisuju tu datoteku. Baket predstavlja kontejner za objekte [6].

3.5. *Amazon RDS*

Amazon Relational Database Service (*RDS*) predstavlja servisnu relacionu bazu podataka koju obezbeđuje *AWS*. *RDS* je zadužen za većinu administrativnih zadataka upravljanja bazom podataka [7].

3.6. *Amazon DynamoDB*

Amazon DynamoDB je *serverless*, *NoSQL* servisna baza podataka. *DynamoDB* pruža izuzetno brzo i perzistentno skladište podataka (engl. *datastore*) zasnovano na modelu ključ-vrednost [8].

3.7. *Amazon VPC*

Amazon Virtual Private Cloud (*VPC*) je servis koji omogućuje pokretanje *AWS* resursa u logički izolovanoj virtuelnoj mreži. Korisnici imaju potpunu kontrolu nad svojim virtuelnim mrežnim okruženjem, uključujući izbor sopstvenog opsega *IP* adresa, kreiranje podmreža (engl. *subnets*), kao i konfiguraciju tabela rutiranja i mrežnih prolaza (engl. *gateways*) [9].

3.8. *Amazon CloudFront*

Amazon CloudFront je veb-servis koji ubrzava distribuciju statičkog i dinamičkog veb-sadržaja aplikacije poput *.html*, *.css*, *.js* i slikovnih datoteka krajnjim korisnicima. *CloudFront* takav sadržaj isporučuje putem svetske mreže data centara, koji se nazivaju ivične lokacije (engl. *edge locations*). Kada korisnik zatraži sadržaj koji se servira preko *CloudFront*-a, zahtev se usmerava na ivičnu lokaciju koja pruža najmanju latenciju (odn. vremensko kašnjenje) kako bi se sadržaj isporučio sa što boljim performansama [10].

3.9. *AWS Secrets Manager*

AWS Secrets Manager je servis koji omogućuje upravljanje, preuzimanje i rotaciju kredencijala za baze podataka i aplikacije, *OAuth* tokena, *API* ključeva i drugih tajnih vrednosti tokom njihovog životnog ciklusa [11].

3.10. *AWS IAM*

AWS IAM (*Identity and Access Management*) je servis koji omogućava upravljanje pristupom *AWS* resursima. Osnovne komponente *IAM*-a su: korisnici (engl. *users*), grupe (engl. *groups*), uloge (engl. *roles*) i polise (engl. *policies*) [12].

3.11. *AWS CDK*

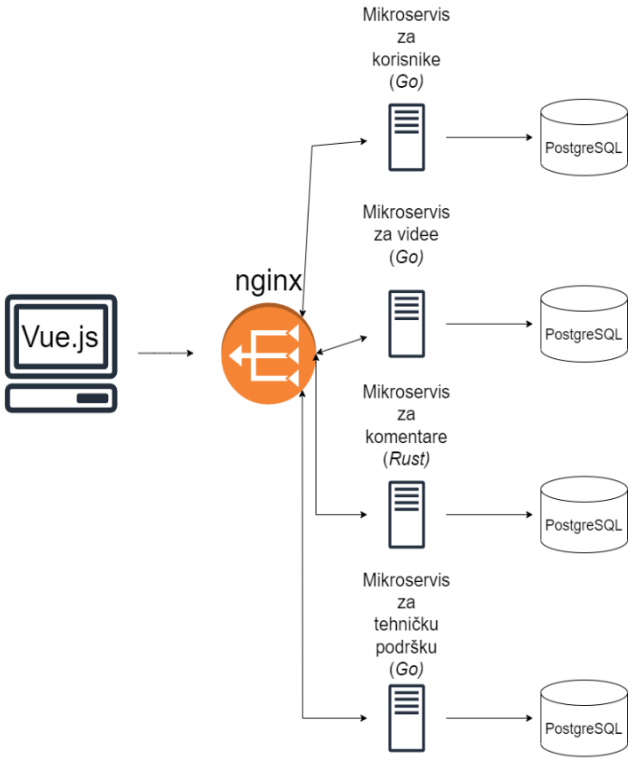
AWS Cloud Development Kit (*AWS CDK*) je radni okvir otvorenog koda (engl. *open-source framework*) za razvoj softvera koji omogućuje definisanje *cloud* infrastrukture putem koda (*IaC*) i njenu implementaciju preko *AWS CloudFormation* servisa. *CDK* pruža objektno-orijentisane apstrakcije visokog nivoa za definisanje *AWS* resursa i servisa na imperativan način, koristeći prednosti savremenih programskih jezika [13].

3.12. *Serverless Framework*

Serverless Framework je radni okvir odn. alat komandne linije koji pojednostavljuje postavku serverles aplikacija i njihove *cloud* infrastrukture kroz pristupačnu *YAML* sintaksu (deklarativni *IaC*) [14].

4. REŠENJE

Na slici 1. prikazana je arhitektura sistema u implementaciji platforme za strimovanje video-sadržaja iz [1]. Glavne komponente su mikroservisi za korisnike, videe, komentare i tehničku podršku, *API Gateway* u vidu *nginx* servera, *PostgreSQL* baze podataka, kao i klijentska *Vue.js* aplikacija.

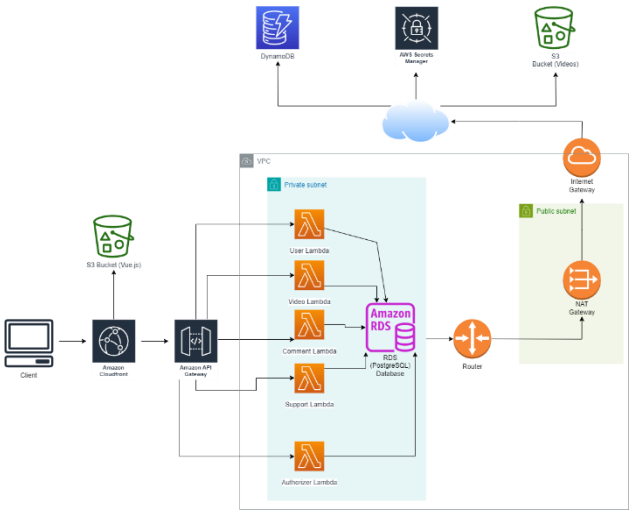


Slika 1. *Arhitektura sistema [1]*

Na slici 2. može se videti arhitektura nakon proširenja, odnosno arhitektura aplikacije na *AWS* infrastrukturi u oblaku, koja je znatno promenjena u odnosu na prethodno rešenje. Celokupan sistem sada je organizovan oko mikroservisne serverles arhitekture.

Glavne komponente nove arhitekture uključuju:

- **Mikroservisi za korisnike, videe, komentare i tehničku podršku** – Svaki od ovih mikroservisa sada je implementiran kao *AWS Lambda* funkcija.



Slika 2. *Arhitektura sistema u AWS oblaku*

- ***AWS API Gateway*** – Za razliku od prethodne arhitekture, koja je koristila *nginx*, sada je primenjen *AWS API Gateway* servis, koji služi kao ulazna tačka za sve *HTTP* i *WebSocket* zahteve ka *Lambda* funkcijama. Ovaj servis takođe obezbeđuje rute pomoću *Lambda* funkcije za autorizaciju, koja je napravljena na osnovu mikroservisa za korisnike.
- **Skladištenje video-sadržaja na *S3*** – Umesto čuvanja na lokalnim diskovima, sada se svi video-zapisi (i *thumbnail*-ovi) čuvaju u *S3* baketu, što omogućuje visoku dostupnost i otpornost na kvarove.
- **Baza podataka** – Svi mikroservisi koriste *Amazon RDS* sa *PostgreSQL* bazom podataka, u kojoj postoji po tabela za svaki entitet kojem odgovara dati mikroservis.
- ***Amazon DynamoDB*** – Ovaj *NoSQL* servis se koristi za specifični slučaj upravljanja *WebSocket* konekcijama u mikroservisu za tehničku podršku, gde su potrebne performanse blizu realnom vremenu.
- ***AWS Secrets Manager*** – U ovom servisu se bezbedno čuvaju kredencijali za *RDS* bazu podataka, *API* ključ i tajni ključ za potpisivanje *JWT* tokena.
- ***VPC* (*Virtual Private Cloud*)** – Sistem je konfigurisan u okviru virtuelne privatne mreže (*VPC*) koja obezbeđuje privatnost i bezbednost komunikacije *Lambda* funkcija sa ostalim servisima i sa internetom. Funkcije i *RDS* baza podataka su smešteni u privatne podmreže radi zabrane dolaznog saobraćaja sa interneta, dok se korišćenje usluga koje su izvan *VPC*-a poput *S3*, *Secrets Manager* i *DynamoDB* ostvaruje putem *NAT* (*Network Address Translation*) *Gateway*-a.
- ***Amazon CloudFront*** – Za distribuciju *Vue.js* aplikacije koristi se *CloudFront* mreža, koja je brzo i pouzdano isporučuje globalno.

Infrastruktura ovog sistema je definisana i postavljena u *AWS* oblak korišćenjem ranije opisanih *AWS CDK* i *Serverless Framework*-a, i jedne *Makefile* skripte.

4.1. *AWS CDK*

AWS CDK iskorišćen je za definisanje putem koda nekoliko delova infrastrukture u vidu *CDK* stekova (koji na *AWS*-u predstavljaju *CloudFormation* stekove): *DynamoDB*, *RDS*, *VPC*, *WebSocket API* stekovi, kao i stek za *deployment* klijentske aplikacije na *AWS* upotrebom *S3* i *CloudFront* servisa. Stekovi su definisani u programskom jeziku *TypeScript*.

Nakon što su definisani, stekovi su postavljeni na *AWS* jednostavnim pozivom komande u komandnoj liniji.

4.2. *Serverless Framework*

Upotrebom *Serverless Framework*-a, odnosno njegovog *YAML* konfiguracionog fajla (*serverless.yml*), definisane su *AWS Lambda* funkcije, njihova konfiguracija, varijable okruženja na osnovu izlaza iz *CloudFormation* stekova (koji postanu dostupni kad se završi *deployment CDK* stekova), uloga i dozvole (permisije), događaji (*HTTP* i *WebSocket*), autorizacija, *CORS* dozvole i slično. Uz *Lambda* funkcije, definisan je i sam *AWS API Gateway*, kao i njegov *API* ključ.

4.3. *Makefile* skripta

Napisana je *Makefile* skripta koja takođe igra ulogu u orkestraciji infrastrukture sistema. Naime, skripta najpre generiše novi tajni ključ za potpisivanje *JWT* tokena i čuva ga u *AWS Secrets Manager*-u upotrebom alata komandne linije za rad sa *AWS*-om: *AWS CLI*. Zatim kreira *S3* baket za video-sadržaj, nakon čega bilduje (engl. *build*) *Lambda* funkcije definisane u *serverless.yml*. Na kraju pokreće komandu za *deployment* konfiguracije *Serverless Framework*-a, što je poslednji korak u postavljanju čitave infrastrukture sistema u *AWS* oblak.

5. ZAKLJUČAK

Ovaj rad predstavlja dalji razvoj platforme za strimovanje video-sadržaja, čija je osnova postavljena u okviru diplomskog rada zasnovanog na mikroservisnoj arhitekturi. Dok je diplomski rad obrađivao mikroservisni pristup u lokalnom okruženju, ovaj rad se fokusira na primenu serverles arhitekture u *AWS* oblaku kao sledećeg koraka u razvoju sistema.

Uvođenje *AWS* servisa, kao što su *Lambda*, *API Gateway* i *S3*, rezultiralo je mogućnošću automatskog prilagođavanja kapaciteta sistema u zavisnosti od opterećenja, čime je sistem postao otporniji na velike fluktuacije u broju korisnika i zahteva. Ovaj pristup eliminiše potrebu za ručnim upravljanjem serverima, što zahteva dodatne resurse i vreme, dok serverles arhitektura omogućava plaćanje isključivo za korišćene resurse, čime se značajno smanjuju troškovi na duži rok u većini primena.

Pored toga, ovakav sistem je jednostavniji za proširivanje novim funkcionalnostima, jer serverles arhitektura podstiče upotrebu malih, nezavisnih komponenti koje se mogu lako integrisati u postojeći sistem bez značajnih izmena. Stoga uvođenje novih funkcija, poput editovanja video sadržaja i optimizacije video-zapisa upotrebom kompresionih algoritama, postaje mnogo efikasnije i lakše za implementaciju, što ujedno predstavlja mogući dalji korak u unapređenju platforme.

6. LITERATURA

[1] Milovan Milovanović, „Platforma za gledanje i deljenje video-sadržaja bazirana na mikroservisnoj arhitekturi“, 2022. , poslednji pristup 27.9.2024.

[2] https://cloud.google.com/learn/what-is-cloud-computing, poslednji pristup 27.9.2024.

[3] https://cloud.google.com/discover/what-is-serverless-architecture, poslednji pristup 27.9.2024.

[4] https://www.serverless.com/aws-lambda, poslednji pristup 27.9.2024.

[5] https://aws.amazon.com/api-gateway/, poslednji pristup 27.9.2024.

[6] https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html, poslednji pristup 27.9.2024.

[7] https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html, poslednji pristup 27.9.2024.

[8] https://en.wikipedia.org/wiki/Amazon_DynamoDB, poslednji pristup 27.9.2024.

[9] https://aws.amazon.com/vpc/features/, poslednji pristup 27.9.2024.

[10] https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Introduction.html, poslednji pristup 27.9.2024.

[11] https://docs.aws.amazon.com/secretsmanager/latest/userguide/intro.html, poslednji pristup 27.9.2024.

[12] https://www.datacamp.com/tutorial/aws-identity-and-access-management-iam-guide, poslednji pristup 27.9.2024.

[13] https://github.com/aws/aws-cdk, poslednji pristup 27.9.2024.

[14] https://github.com/serverless/serverless, poslednji pristup 27.9.2024.

Kratka biografija:

Milovan Milovanović rođen je u Novom Sadu 1999. god. Diplomski rad na Fakultetu tehničkih nauka u Novom sadu iz oblasti Elektrotehnike i računarstva – Softversko inženjerstvo i informacione tehnologije odbranio je 2022. god.

kontakt: milovanovicm309@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.738.5:339
DOI: https://doi.org/10.24867/31BE03Stamenkovic

SKLADIŠTENJE I RAZMENA ZDRAVSTVENIH PODATAKA PRIMENOM BLOCKCHAIN TEHNOLOGIJE

STORAGE AND EXCHANGE OF HEALTH DATA USING BLOCKCHAIN TECHNOLOGY

Lea Stamenković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Rad istražuje primenu blockchain tehnologije u zdravstvu, sa posebnim osvrtom na razmenu i sigurnost podataka o pacijentima. U radu su analizirani problemi sa kojima se suočava zdravstvena industrija, predstavljena su dosadašnja rešenja i objašnjena ključna terminologija. Istaknute su prednosti koje blockchain donosi u ovoj oblasti, uključujući decentralizaciju i sigurnost. Tehnologija blockchain-a je detaljno razmatrana kroz opis njenih elemenata, načina funkcionisanja, kao i izazova u primeni. Rad obuhvata i opis implementacije rešenja zasnovanog na blockchain-u, koristeći IPFS za skladištenje podataka, Spring Boot za back-end, te Angular za front-end aplikaciju.*

Ključne reči: *Medicina, EHR, Blockchain, Hyperledger Fabric, razmena zdravstvenih podataka*

Abstract – *The paper explores the application of blockchain technology in healthcare, with a focus on the exchange and security of patient data. It analyzes the challenges faced by the healthcare industry, presents existing solutions, and explains key terminology. The advantages that blockchain brings to this field, including decentralization and security, are highlighted. The technology is thoroughly examined through a description of its elements, functioning, and the challenges of its implementation. The paper also includes a description of a blockchain-based solution, using IPFS for data storage, Spring Boot for the back-end, and Angular for the front-end application.*

Keywords: *Medicine, electronic health records (EHR), blockchain, Hyperledger Fabric, health data storage and exchange*

1. UVOD

Ključni deo svake medicinske ustanove čini medicinska dokumentacija. Mnoge organizacije, među kojima je i Svetska Zdravstvena Organizacija (WHO), decenijama u nazad rade na sistematizaciji samog procesa kreiranja medicinskih dokumenata, u cilju smanjenja broja nedostajućih i netačnih podataka u njima. Mogućnost da zdravstveni radnici digitalno čuvaju podatke o pacijentu uz obezbeđenu sigurnost tih podataka u mnogome je doprinela povećanju efikasnosti i smanjenju troškova. U proteklim decenijama, EHR sistemi su postali ključni za savremenu zdravstvenu zaštitu, ali uprkos njihovoj širokoj upotrebi, određeni izazovi i dalje postoje. Interoperabilnost se navodi kao jedan od značajnih problema EHR sistema koji su trenutno u upotrebi [1]. Takođe, skorašnje studije pokazuju da je broj povreda u značajnom porastu u proteklih par godina, gde se ističu hakovanje, IT incidenti i neovlašćeni pristup kao najčešći načini povrede podataka [2]. Povrede podataka naglašavaju rastuću ranjivost osetljivih informacija, ali takođe pokazuju značajne bezbednosne izazove i izazove privatnosti EHR sistema. Za rešavanje pomenutih problema, sve više pažnje privlači blockchain tehnologija [3]. Zbog svoje pouzdanosti i interoperabilnosti, blockchain pruža bezbednu infrastrukturu za deljenje EHR podataka između ovlašćenih strana. Uz pomoć pametnih ugovora i mehanizama konsenzusa između učesnika u mreži, blockchain omogućuje potpunu kontrolu nad pristupom podacima. Time zasigurno obezbeđuje povećanu sigurnost i privatnost kao osnovnu prednost primene. Uklanjanjem centralizovanih odluka i oslanjanjem na kriptografske algoritme, blockchain značajno smanjuje rizik od neovlašćenog pristupa i povrede podataka. Dodatno, blockchain možemo posmatrati i kao okvir (framework), koji omogućava nesmetanu razmenu podataka između različitih entiteta u zdravstvu, nezavisno od sistema koje oni trenutno koriste [4].

2. KRIPTOGRAFIJA

Kriptografija predstavlja metod zaštite podataka putem kodiranja, kako bi se sprečio pristup neovlašćenim licima. Ona igra ključnu ulogu u blockchain, gde obezbeđuje sigurnost transakcija, autentifikaciju korisnika i integritet podataka. Blockchain koristi kriptografske heševe za povezivanje blokova, čime se osigurava da podaci u lancu ne mogu biti promenjeni bez otkrivanja.

2. BLOCKCHAIN

Blockchain [5] je distribuirana baza podataka koja čuva informacije o transakcijama među entitetima bez centralnog autoriteta. Održavaju je članovi mreže, a svaka nova transakcija se vezuje za prethodnu, čime nastaje lanac blokova. Zahvaljujući kriptografskim heš funkcijama, podaci se ne mogu menjati bez saglasnosti većine čvorova, čime se obezbeđuje integritet i

---

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

nepromenljivost podataka. Mreža je peer-to-peer, što znači da su svi čvorovi ravnopravni i mogu direktno komunicirati, čime se povećava otpornost na kvarove. Iako se podaci mogu dodavati, njihovo menjanje zahteva velike resurse i saglasnost više od 50% čvorova. U suštini, blockchain funkcioniše kao sigurni distribuirani registar koji omogućava transparentnost i poverenje među korisnicima.

3. HYPERLEDGER FABRIC

Hyperledger Fabric (HLF) [6] je modularni blockchain radni okvir koji omogućava razvoj rešenja i aplikacija zasnovanih na blockchain tehnologiji uz korišćenje modularnih komponenata. Radi implementacije ovakve arhitekture, HLF sadrži modularne blokove: ordering servis, Membership service provajder, peer-to-peer protokol za prosleđivanje informacija, pametne ugovore, glavnu knjigu. **Glavnu knjigu** u HLF blockchain-u čine stanje i blockchain. Stanje (world state) predstavlja bazu podataka koja čuva trenutne vrednosti stanja na blockchain-u. Blockchain je zapis svih transakcija koje su dovele do trenutnog stanja. Ove transakcije su organizovane u blokove povezane u lanac. Za razliku od stanja, podaci u blockchain-u se ne mogu naknadno menjati. **Pametan ugovor** je kod – koji poziva aplikacija klijenta izvan mreže blockchain-a – koji upravlja pristupom i izmenama skupa ključ-vrednost u glavnoj knjizi (tj. world state). U HLF, pametni ugovori nazivaju se chaincode. Chaincode pametnog ugovora instalira se na čvorove i inicijalizuje na jedan ili više kanala. **Ordering servis** predstavlja skup čvorova koji slažu transakcije u blok. Funkcioniše nezavisno od procesa čvorova i slaže transakcije na osnovu principa "prvi dođe, prvi uslužen" za sve kanale na mreži. Ordering servis je zajednički za celu mrežu i sadrži kriptografske identitete povezane sa svakim članom. **MSP** je zadužen za autentifikuju, autorizuju i upravljaju identitetima na blockchain mreži, ali i autentifikuju i autorizuju i blockchain operacije. On predstavlja apstraktnu komponentu sistema koji pruža akreditive klijentima i čvorovima kako bi učestvovali u HLF mreži. Klijenti koriste ove akreditive za autentifikaciju svojih transakcija, a čvorovi koriste ove akreditive za autentifikaciju rezultata obrade transakcija (*endorsement*).

4. BLOCKCHAIN I ZDRAVSTVO

Zdravstvena industrija suočava se sa nizom specifičnih potreba, naročito u pogledu sigurnosti, privatnosti podataka i efikasnog upravljanja informacijama. Osetljivost medicinskih podataka nameće stroge regulatorne zahteve, što značajno otežava njihovu razmenu između različitih aktera u zdravstvu. Pored toga, trenutni sistemi za upravljanje medicinskim podacima često su decentralizovani i fragmentirani, što dodatno komplikuje efikasan protok informacija.

Zdravstveni sistemi imaju potrebu za tehnologijom koja može povezati različite delove industrije, omogućiti verifikaciju podataka, ali i garantovati integritet i nepovredivost informacija. Blockchain nudi infrastrukturnu osnovu koja može podržati sve ove zahteve, pružajući okvir u kojem su informacije lako dostupne ovlašćenim korisnicima, a istovremeno zaštićene od neovlašćenih pristupa ili manipulacija. Primena blockchain tehnologije u zdravstvu polako postaje sve veća. Prema skorašnjim predviđanjima, globalno tržište blockchain-a u zdravstvu očekuje skok od 23-60%, do 2030. godine. Ovaj rast najlakše je objasniti razvojem blockchain tehnologije kroz različite generacije, gde se za generaciju X ističe korišćenje blockchain-a u svakodnevnom životu kroz veštačku inteligenciju.

3.1 Relevantni radovi

MedRec [7] je blockchain sistem za upravljanje medicinskim podacima, razvijen za poboljšanje sigurnosti i kontrole pristupa zdravstvenim informacijama koristeći Ethereum mrežu. Sistem omogućava pacijentima da kontrolišu ko ima pristup njihovim podacima putem pametnih ugovora na blockchain-u. Ovi ugovori omogućavaju odobravanje ili odbijanje pristupa zdravstvenim radnicima i institucijama. MedRec implementira tri vrste ugovora: Ugovor o Registraciji (RC), koji preslika identifikacione stringove na Ethereum adrese i reguliše registraciju novih identiteta; Ugovor o Odnosima Pacijent-Pružalac (PPR), koji upravlja medicinskim zapisima između pacijenata i pružalaca usluga; i Ugovor o Rezimeu (SC), koji sadrži istoriju medicinskih zapisa. Sistem se integriše sa postojećom infrastrukturom za elektronske medicinske zapise (EMR) kroz četiri softverske komponente: Backend Biblioteku, Ethereum Klijenta, Čuvara Baze Podataka i EMR Menadžera. MedRec koristi Proof of Work (PoW) kao osnovni konsenzusni algoritam, ali uvodi dodatne modele motivacije za rudarstvo specifične za zdravstveni sektor.

UniRec (Unified Medical Records) [8] sistem funkcioniše kao privatna peer-to-peer (P2P) mreža koju dele različite zdravstvene organizacije. U ovoj mreži, blockchain Ethereum održava zajedničku istoriju svakog EHR-a, dok se medicinski podaci razmenjuju između institucija preko IPFS-a. UniRec koristi Node.js za upravljanje blockchain-om i šifrovanje sadržaja, dok pametni ugovori omogućavaju kontrolu pristupa.

MedicalChain [9] je platforma koja omogućava pacijentima da kontrolišu pristup svojim podacima putem pametnih ugovora, dok se transakcije beleže na blockchain-u. MedicalChain ima sistem plaćanja koristeći tokene (MedTokens), koje pacijenti mogu koristiti za različite usluge. Model se sastoji od dvostruke blockchain strukture, pri čemu jedna kontroliše pristup zdravstvenim zapisima, a druga služi kao osnova za aplikacije i usluge platforme.

4. MODEL SISTEMA

4.1. Arhitektura sistema

Sistem je dizajniran korišćenjem savremenih tehnologija, uključujući Angular za front-end, Spring za back-end, Hyperledger Fabric za blockchain infrastrukturu i IPFS za skladištenje datoteka. Ova arhitektura omogućava i sigurno i efikasno upravljanje podacima pacijenata.

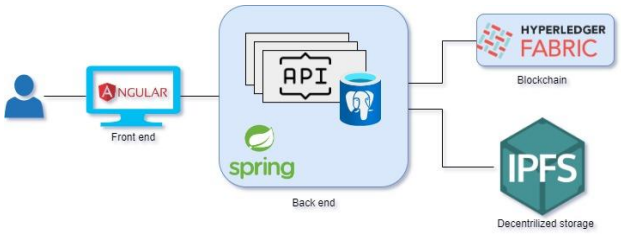
Angular aplikacija omogućava korisnicima interakciju putem web pretraživača, s intuitivnim korisničkim interfejsom za pregled i ažuriranje pacijentskih kartona. Koristi REST API-je za komunikaciju sa Spring back-end aplikacijom.

Spring back-end upravlja poslovnom logikom, autentifikacijom i autorizacijom, kao i obradom podataka. Uključuje API-je za interakciju sa Hyperledger Fabric i upravljanje IPFS-om za skladištenje dokumenata, kao i REST API za komunikaciju sa front-end-om.

Hyperledger Fabric pruža privatni blockchain okvir za sigurno upravljanje podacima. Pametni ugovori upravljaju poslovnim pravilima i verifikacijom transakcija, osiguravajući nepromenljivost i sigurnost podataka.

IPFS omogućava distribuirano skladištenje velikih datoteka povezanih sa pacijentskim kartonima. Dokumenti se skladište na IPFS-u, dok se veze do njih čuvaju na blockchain-u, olakšavajući preuzimanje i deljenje uz očuvanje integriteta.

Arhitektura sistema prikazana je na slici 1.



Slika 1. *Arhitektura rešenja*

4.2 Funkcionalnosti sistema

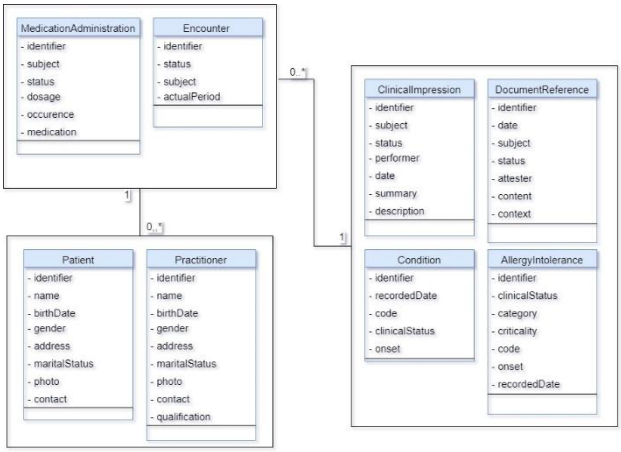
U sistemu su identifikovani sledeći akteri: lekari, pacijenti i administratori. Lekari koriste sistem za pregled pacijenta, pristup i ažuriranje pacijentskih kartona. Pacijenti imaju mogućnost da pregledaju svoje kartone, ažuriraju osnovne podatke i pregledaju logove pristupa svom kartonu. Administratori su odgovorni za upravljanje korisničkim nalozima. Sva tri tipa korisnika imaju mogućnost da menjaju podatke koji se eksplicitno tiču aplikacije (imejl, lozinka,..). Takođe, prepoznajemo i neregistrovanog pacijenta kao deo sistema, kom je omogućeno da se registruje, čime će, ukoliko njegov karton ne postoji već u sistemu, isti biti i kreiran. Ukoliko neregistrovan pacijent već ima postojeći karton u sistemu, ali nije deo aplikacije, kreiraće se novi nalog u okviru aplikacije koji će biti povezan sa njegovim postojećim kartonom.

4.3. Model podataka

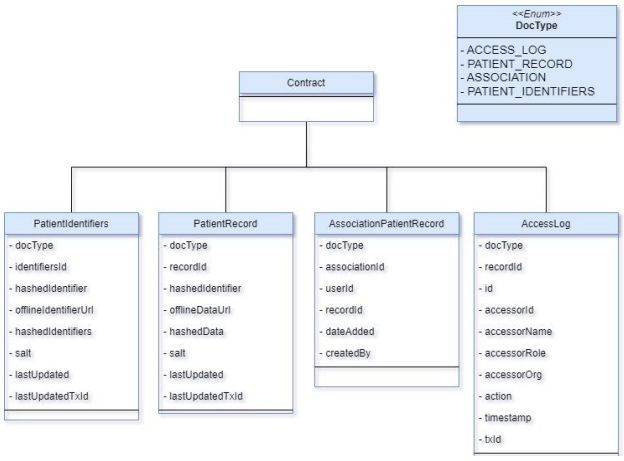
Back-end Spring aplikacija čuva informacije o korisniku, sa atributima koji ga definišu. Svaki korisnik ima podatke potrebne za autentifikaciju (email, password) i osnovne informacije (ime, prezime, imagePath).

Podaci o pacijentskom kartonu čuvaju se van blockchain mreže. Oni predstavljaju skup HAPI FHIR [10] resursa. Dijagram koji prikazuje polja off-chain podataka prikazan je na slici 2.

Podaci na mreži, prikazani na slici 3, služe kako bi se odredila prava pristupa podacima, povezao korisnik platforme sa njegovim kartonom, kao i da bi se proverio integritet podataka sačuvanih u off-line bazi. Model AssociationPatientRecord, omogućava povezivanje objekta PatientIdentifiers i PatientRecord. Ovakva implementacija omogućava da se na osnovu identifikatora kojeg korisnik koristi u zdravstvenim ustanovama poveže karton pacijenta sa već postojećim u sistemu. Ove informacije nalaze se unutar PatientRecord. Međutim dalje u komunikaciji se ne koristi njegov identifikator, već identifikator iz same aplikacije, što se čuva u PatientIdentfiers. Ovaj objekat skuplja sve identifikatore jednog korisnika koje on ima potencijalno u više zdravstvenih institucija. Model AccessLog skladišti sve pristupe pacijentskim kartonima.



Slika 2. *Model podataka u bazi van mreže*



Slika 3. *Model podataka unutar mreže*

5. IMPLEMENTACIJA

5.1 FHIR

Jedan od zahteva zadatka bio je i omogućiti nesmetanu razmenu pacijentskih kartona između različitih institucija. Da bi ovo bilo moguće, korišćen je standard FHIR i njegova Java implementacija HAPI FHIR *(Health API Fast Healthcare Interoperability Resources*). U ovom projektu korištena je samo radi strukturiranja podataka, u skladu sa standardom koji je dobro poznat u zdravstvu.

5.2 IPFS

IPFS (*InterPlanetary File System*) je distribuirani fajl sistem zasnovan na peer-to-peer mreži. Adresiranje u ovom fajl sistemu je zamenjeno adresiranjem zasnovanim na sadržaju. Drugim rečima, za pretragu nekih podataka potreban je njihov heš, a ne adresa na kojoj se nalaze. Kada se fajl pošalje na IPFS radi skladištenja, generiše se jedinstveni heš za taj fajl. Stoga, za pronalaženje tog fajla, potrebno je pretražiti njegov heš. Zbog decentralizacije, očuvanja privatnosti i nepromenljivosti podataka na IPFS-u, ovaj fajl sistem je korišten u projektu kao off-line baza, na kojoj su skladišteni pacijentski kartoni.

5.3 Blockchain

Blockchain mreža ovog projekta razvijena je korišćenjem Hyperledger Fabric (HLF), koji implementira privatnu permissioned blockchain mrežu. Ovaj tip mreže je izuzetno pogodan za primenu u zdravstvu, gde je ključno ograničiti pristup informacijama samo na entitete koji pripadaju zdravstvenim institucijama ili su njihovi korisnici. Pristup mreži zahteva posedovanje sertifikata koji dokazuje identitet korisnika, a upravljanje identitetima je u potpunosti podržano unutar sistema.

U implementaciji je korišćena testna mreža, detaljno opisana u dokumentaciji Hyperledger Fabric-a [6], koja predviđa postojanje dve organizacije uz mogućnost uključivanja treće. Između ovih organizacija kreiran je kanal koji omogućava sigurnu razmenu informacija. Mreža je takođe postavljena uz korišćenje infrastrukture ključeva, što znači da svaka organizacija ima svoj CA (Certificate Authority) koji potpisuje sertifikate koje izdaju te organizacije.

Postizanje dogovora u HLF mreži zavisi od politike odobravanja i pametnog ugovora. Politika odobravanja definiše ko ima pravo da odobri ili odbije određene radnje. U testnoj mreži, politika odobravanja konfigurisana je tako da zahteva odobrenje od bilo kog čvora kanala. Pametan ugovor se instalira na kanal, i svaki čvor organizacije koja je deo kanala. Pametan ugovor omogućava granuliranu kontrolu pristupa resursima na mreži, i enkapsulira poslovnu logiku. U ovom slučaju, pametan ugovor vrši provere uloge korisnika sistema i na osnovu tih provera odobrava određene akcije nad podacima, čime se obezbeđuje da samo ovlašćeni korisnici mogu izvršavati kritične operacije. Primer funkcije pametnog ugovora u kojoj se dobavlja pacijentski karton prikazan je u listingu 1.

```
async GetPatientRecord(ctx, hashedUserId,time) {
   let role = await Util.GetUserRole(ctx);
   if(role !== 'ROLE_PRACTITIONER'){
      throw new Error('unauthorized access to
patient record!'); }
   let association = await
AssociationPatientRecordChaincode.GetAssociation
(ctx, hashedUserId);
   if(!association){
      throw new Error('Invalid patient id.'); }
   let patientRecord = await
PatientRecordChaincode.GetPatientRecord(ctx,
association.recordId);
   await AccessLogChaincode.AddAccessLog(ctx,
association.recordId, time, Action.VIEW);
   return JSON.stringify(patientRecord); }
```

Listing 1. – *Primer funkcije pametnog ugovora*

6. ZAKLJUČAK

U ovoj studiji istraživana je primena blockchain tehnologije u razmeni i čuvanju pacijentskih kartona, s fokusom na koristi i izazove u zdravstvu.

Blockchain nudi značajne prednosti u sigurnosti, transparentnosti i integritetu podataka. Elektronski zdravstveni zapisi (EHR) su ključni za efikasno upravljanje informacijama, a korišćenje blockchain-a omogućava sigurno čuvanje i razmenu podataka među različitim entitetima u zdravstvenom sistemu.

Tehnologija donosi revoluciju u čuvanju i razmeni podataka, nudeći poboljšanja u bezbednosti i kontroli pristupa. U budućnosti, moguća su dalja unapređenja poput kreiranja više kanala unutar blockchain mreže sa različitim pametnim ugovorima, kao i povećane kontrole pristupa koju definišu sami pacijenti, što bi poboljšalo privatnost i sigurnost podataka. Uvođenje uređaja za unos informacija u realnom vremenu moglo bi dodatno unaprediti tačnost podataka.

7. LITERATURA

[1] Li, Edmond, et al. "Electronic health records, interoperability and patient safety in health systems of high-income countries: a systematic review protocol." BMJ open 11.7 (2021): e044941.

[2] Clement, Tosin, et al. "Cyber Analytics: Modelling the Factors Behind Healthcare Data Breaches for Smarter Security Solutions." International Journal of Advance Research, Ideas and Innovations in Technology 10.1 (2024): 49-75.

[3] Bashir, Imran. Mastering blockchain. Packt Publishing Ltd, 2017.

[4] RASEL, MD, and Revathi Bommu. "Blockchain-Enabled Secure Interoperability: Advancing Electronic Health Records (EHR) Data Exchange." International Journal of Advanced Engineering Technologies and Innovations 1.3 (2024): 262-281.

[5] Sarmah, Simanta Shekhar. "Understanding blockchain technology." Computer Science and Engineering 8.2 (2018): 23-29.

[6] Hyperledger Fabric docs., [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.5. [Accessed 9 2024].

[7] Azaria, Asaph, et al. "Medrec: Using blockchain for medical data access and permission management." 2016 2nd international conference on open and big data (OBD). IEEE, 2016.

[8] Quaini, Tiago, et al. "A MODEL FOR BLOCKCHAIN-BASED DISTRIBUTED ELECTRONIC HEALTH RECORDS." IADIS International Journal on WWW/Internet 16.2 (2018).

[9] Capece, Guendalina, and Francesco Lorenzi. "Blockchain and Healthcare: Opportunities and Prospects for the EHR." *Sustainability* 12.22 (2020): 9693.

[10] HAPI FHIR, [Online]. Available: https://hapifhir.io/. [Accessed 9 2024].

Kratka biografija:

Lea Stamenković je rođena 26. 6. 1999. u Subotici. Osnovne akademske studije je završila 2022. godine na Fakultetu tehničkih nauka u Novom Sadu. Master rad na Fakultetu tehničkih nauka iz oblasti Računarstva i automatike – Elektronsko poslovanje odbranila je 2024. godine.
kontakt: lea.stamenkovic99@gmail.com

Зборник радова Факултета техничких наука, Нови Сад

УДК: (Уписује редакција, оставити овај ред)
ДОИ: (Уписује редакција, оставити овај ред)

ДЕТЕКЦИЈА ЕМОЦИЈА ОПТИЧКИМ ПУТЕМ АНАЛИЗОМ ФАЦИЈАЛНИХ ЕКСПРЕСИЈА

OPTICAL METHOD FOR EMOTION DETECTION BY ANALYZING FACIAL EXPRESSIONS

Немања Томовић, др Владимир Рајс, Факултет техничких наука, Нови Сад

Област – ЕЛЕКТРОНИКА И РАЧУНАРСТВО

Кратак садржај – *У овом раду приказана су психолошка знања и истраживања из области људских емоција, при чему се ове информације користе како би се пронашло техничко решење проблема детекције емоција на основу фацијалне експресије. Рад обрађује више метода, њихове предности и недостатке; имплементирајући методу која помоћу камере и алгоритма машинског учења перципира емоције.*

Кључне речи: *емоције, физички и физиолошки процеси, сензори, машинско учење, неуронске мреже, raspberry pi, python, tensorflow, keras*

Abstract – *In this paper knowledge and researches gained from the field of psychology of human emotion are shown alongside the important problem solving implementation-necessary knowledge. The paper processes multiple methods, as well as the advantages and disadvantages; implementing the method which provides solution to the problem using camera module and machine learning algorithm.*

Keywords: *emotion, physical and physiological processes, sensors, machine learning, neural networks, raspberry pi, python, tensorflow, keras*

1. УВОД

Емоције су компликовани процес који се одиграва у телу појединца. У данашње време наука добро познаје основне принципе настанка емоција, али недовољно добро да би их описала математичким алатом [1]. Сходно томе, алгоритам машинског учења је идеалан за решавање поменутог типа проблема. Неопходно је изазвати појаву емоције код испитаника, у контролисаним условима и сачувати те информације које ће касније бити кључне за обуку неуронских мрежа. У овом раду истражени су принципи детекције биосигнала и телесних експресија услед промене емотивног стања. Фокус имплементације је постављен на детекцији телесних промена (фацијалних промена, тзв. фацијалних експресија) због своје једноставности, исплативности и практичности. У наставку биће приказана и имплементација једног таквог алгоритма.

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

1.1. Биосигнали

Биосигнали су свеприсутни у људском телу и представљају „одговор“ рада одређеног органа. На пример, срце генерише импулсе које називамо откуцаји срца, док мозак генерише магнетно поље услед електричних активности које се у њему догађају [2]. Поремећај у раду биосигнала је приметан услед промене емотивног стања, те се они могу користити за детекцију емоције. Ипак, метода није практична јер захтева велики број сензора, габаритних је димензија и крута.

1.2. Телесне експресије

Покрети могу бити вољни и безвољни. Вољни покрет је покрет који правимо свесно, на пример писање. Безвољни покрет не можемо свесно да контролишемо и последица је реакције аутономног нервног система. При промени емотивног стања, органи почињу да раде абнормално услед чега се лучи много већи број хормона што се коначно испољава кроз безвољну телесну реакцију (поскакивање, смејање, подрхтавање и слично). Перципирањем телесних експресија може се донети закључак коју емоцију осећа појединац [3]. Недостатак ове методе је што манипулативна понашања, где појединац свесно имитира телесне експресије зарад исказивања емоције коју не осећа, ће преварити систем.

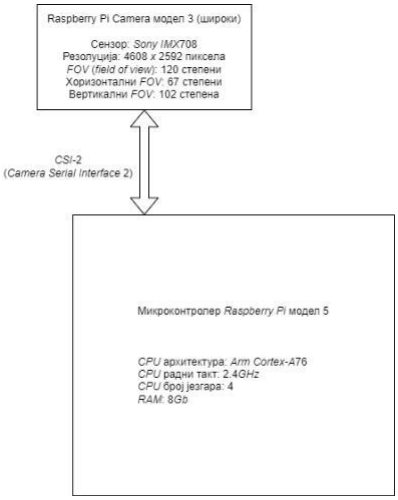
1.3. Перцепција и машинско учење

Временом сваки појединац научи да перципира ствари око себе. Једна од тих ствари су људске емоције. Перцепција се одвија несвесно што се доказује тиме да човек иако може да да тачан одговор, тешко му је да објасни зашто је баш тај одговор дао. Иста ситуација је и код машинског учења. Машини се дају проверени подаци; за које се зна шта представљају. Са порастом примера, машина временом почиње да перципира податке и исправно препознаје шта се на подацима налази. Покушајем анализирања одговора неуронске мреже, која имплементира алгоритам машинског учења, наилази се на зид. Исто као човек, машина није свесна зашто је дала баш тај одговор, али сигурна је у њега. Ипак, постоји мала разлика која разграничава човека од машине. Машинско учење је исказ математичког језика, комплексна функција, те машина увек квантификује сигурност свог одговора (0-100%), док човек, у најбољем случају, то може да учини само описно.

2. ПРАКТИЧНА РЕАЛИЗАЦИЈА

2.1. Реализација на микропроцесорској јединици

На слици 1 приказана је практична реализација проблема коришћењем методе анализе телесних експресија уз ограничење – коришћен је само подскуп информација (фацијална експресија). Овај подскуп даје највише потребних информација за решавање проблема детекције емоционалног стања. Остатак информација, понаособ гледано, нису довољни за решавање проблема, али могу пружити већу сигурност у тачност одговора.



Слика 1: *Блок шема реализације детекције људских емоција анализом фацијалних експресија*

Да би се проблем решио користи се оптички сензор (камера) који прибавља слику и предаје је микропроцесорској јединици (raspberry pi) на даљу обраду. На микроконтролеру налази се обучена неуронска мрежа за решавање проблема класификације емоција. Улазна слика се обрађује и предаје неуронској мрежи. Улазна слика у неуронску мрежу је величине 224x224 пиксела у троканалном RGB (енгл. red green blue, срп. црвена зелена плава) формату.

```
41
42  emotion_recognition_model = tf.keras.models.load_model ("neural_network/v8.h5")
43  face_cascade = cv2.CascadeClassifier (
44      cv2.data.haarcascades +
45      "haarcascade/haarcascade_frontalface_default.xml"
46  )
47
48  cv2.namedWindow ("Full Screen", cv2.WND_PROP_FULLSCREEN)
49  cv2.setWindowProperty ("Full Screen", cv2.WND_PROP_FULLSCREEN, cv2.WINDOW_FULLSCREEN)
50
51  picamera = Picamera ()
52  picamera.configure( picamera.create_preview_configuration ( main = {
53      "format" : "RGB888",
54      "size" : (1024, 600)
55      #"size" : (4608, 2592)
56      }))
57  )
58
59  picamera.start ()
60
```

Код 1: *Конфигурација и креирање потребних објеката за програм за детекцију људских емоција*

Са кода 1 приметно је учитавање модела неуронске мреже за класификацију емоција (линија 42), модела за детекцију људског лица, базираног на Харовој таласној трансформацији (линија 43), као и конфигурација и започињање рада камера модула (линије 52, 59).

```
61  while True:
62      image_RGB = picamera.capture_array ()
63      image_grayscale = cv2.cvtColor (image_RGB, cv2.COLOR_RGB2GRAY)
64      detected_faces = face_cascade.detectMultiScale (
65          image_grayscale,
66          HAARCASCADE_SCALE_FACTOR,
67          HAARCASCADE_MIN_NEIGHBOURS,
68          minSize = (50, 50)
69      )
70
71      roi = find_face_roi (detected_faces, image_RGB, image_grayscale)
72      if (roi is not None):
73          image_nn = cv2.resize (roi, (IMAGE_SIZE, IMAGE_SIZE))
74          #image_nn = image_nn.reshape (1, IMAGE_SIZE, IMAGE_SIZE, 3)
75          image_nn = np.expand_dims (image_nn, axis = 0)
76          image_nn = image_nn / 255.0
77
78          predictions = emotion_recognition_model.predict (image_nn)
79          predicted = np.argmax (predictions)
```

Код 2: *Runtime програма – преузимање улазних информација и обрада, проналазак лица на слици и класификација емоције путем неуронске мреже*

Са кода 2 види се да микропроцесорска јединица (raspberry pi) преузима слику са камере (линија 62). Ова слика се потом обрађује у једноканални монохроматски формат (линија 63) како би се прилагодила облику потребном да Харова таласна трансофмација пронађе локацију лица на слици (линија 64), уколико оно постоји. Лице се изолује од остатка слике (линија 71), величина слике се прилагођава (линија 73) и формат информација прилагођава улазу неуронске мреже за класификацију емоција (линије 75, 76) након чега се шаље упит овој мрежи (линија 78) и анализира њен одговор (линија 79).

```
81          match (predicted):
82              case 0:
83                  emotion = "bes"
84              case 1:
85                  emotion = "gadjenje"
86              case 2:
87                  emotion = "strah"
88              case 3:
89                  emotion = "sreca"
90              case 4:
91                  emotion = "neutralno"
92              case 5:
93                  emotion = "tuga"
94              case 6:
95                  emotion = "iznenadjenje"
96              case _:
97                  emotion = "unknown"
```

Код 3: *Runtime програма – мапирање одговора неуроснке мреже на људски разумљив језик*

Код 3 приказује мапирање одговора неуронске мреже за класификацију емоција у одговор смислен човеку. Наиме, мрежа даје одговор у систему бројевних вредности од 0 до 6 што се даље мапира у људски разумљиве информације:

- бес (0);
- гађење (1);
- страх (2);
- срећа (3);
- неутрално (4);
- туга (5);
- изненађење (6).

Мапирање емоција на број није произвољно изведено. Оно зависи од начина на који је мрежа обучена и шта јој је речено да свака од ових бројних вредности представља. У наредном поглављу ће бити приказана обука неуронске мреже.

2.2. Реализација обуке неуронске мреже

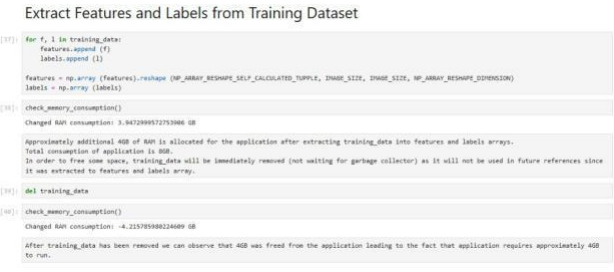
На коду 4 може се видети мапирање означених бројевних вредности излаза неуронске мреже на литералну вредност емоционалне класе коју представљају.

Код 4: *Мапирање излаза на људски читљиву вредност*

Приликом обуке неуронске мреже кориштен је алат *jupyter notebook*. Алат се показао као идеално решење због недовољно *RAM* (енгл. *Random Access Memory*) ресурса. Инструкције су често извршаване у фазама како би се омогућило програму да се успешно изврши. Алгоритми машинског учења су поприлично захтевни за рачунар захтевајући велики меморијски утрошак зарад учитавања података за обуку мреже и математичког прорачуна параметара мреже који се обучавају. Код 5 приказује учитавање целокупног сета за обуку неуронске мреже. Напомиње се да сет за валидацију неуронске мреже није кориштен у овом раду због недостатка меморијског простора.

Код 5: *Учитавање сета за обуку неуронске мреже*

Код 6 приказује екстракцију података из сета за обуку. Подаци из овог сета се раздвајају на слику, која приказује одређену емоцију (*features* објекат), и ознаку, о којој емоцији је реч (*labels* објекат). Са истог кода може се приметити да је у току имплементације праћен утрошак меморије и ресурси су експлицитно ослобађани.



Код 6: *Екстракција сета за обуку у објекат података за обуку и објекат ознака класе за класификацију*

Код 7 приказује прилагоду улазног податка. Прва инструкција означена као [50], је извршена како би се смањила алокација меморије, опет.

Типично, вредности из реалног система се смештају у максималну ширину меморије одређену архитектуром рачунара (у случају кориштеног лаптопа *Victus* ради се о 64-битним меморијским ћелијама). Накнадна инстру-кција представља нормализацију података из [0, 255] (осмобитни *RGB* формат) у [0, 1] (скуп реалних бројева са којима ради мрежа).

Код 7: *Прилагода и нормализација података типа слика*

При обуци кориштен је принцип преносног учења. Изабран је модел *MobileNetV2,* који омогућава класификацију до 1000 различитих класа слика. Овај модел је обучен над подацима типа слике, због чега се за било који проблем тог типа може искористити зарад убрзавања процеса учења. Учитавање је приказано кодом 8. Код 9 приказује прилагоду излаза са 1000 класа на 7 (онолико колико се у овом раду детектује). Додатно, излазни слој није замењен једним од 7 неурона, већ са три слоја са по 128, 64 и 7 неурона (респективно од скривеног слоја ка излазу гледано). Број замењених слојева и број изабраних неурона је произвољно одабран.

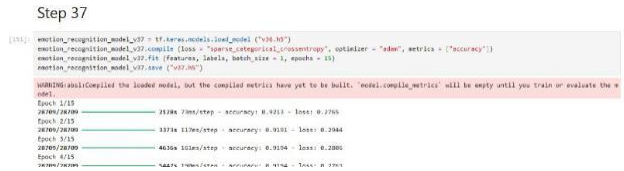
Код 8: *Учитавање претренираног модела за решавање проблема класификације слика MobileNetV2*

Код 9: *Прилагода излазног слоја учитане мреже како би одговарала проблему класификације 7 класа*

Извршне промене су, потом, учитане, што приказује код 10, чиме је формирана неуронска мрежа за решавање проблема класификације емоција.

Код 10: *Учитавање измена у модел неуронске мреже*

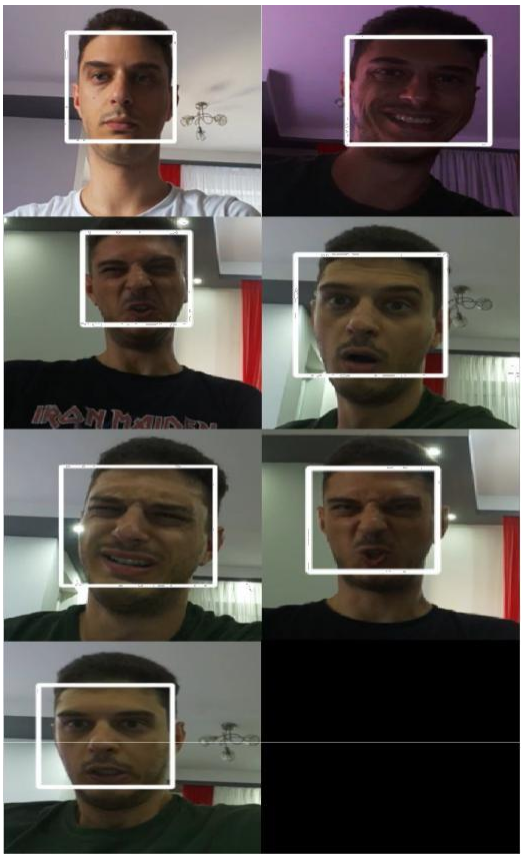
Код 11 приказује један од корака обуке неуронске мреже. Обука је вршена у фазама. Укупно је извршено 30 фаза у којима се узима 1/32 укупног броја података за обуку. Накнадно је обављено још 7 фаза где су сви подаци из мреже за обуку дати одједном.



Код 11: *Приказ једне фазе обуке неуронске мреже*

3. РЕЗУЛТАТИ

У овом одељку дат је приказ неких од добијених резултата имплементираног решења. На слици 2 приказана је по једна слика сваке емоције добијена у различитим оперативним условима. У првој колони се могу видети неутрално стање, гађење, туга и страх, док се у другој колони могу видети срећа, изненађење и бес.



Слика 2: *Приказ резултата имплементиране мреже над свим класама*

4. ЗАКЉУЧАК

Детекција емоција оптичким путем је изузетно практична метода јер не захтева посебно дизајниране круте уређаје габаритних димензија. Довољно је поставити камеру и пустити алгоритам да одради остатак посла. Практична примена ове апликације се изналази само у информативним и забавним сврхама. Унапређење апликације се може одвити у два смера.

У првом смеру апликација би се побољшала додавањем информација о говору тела (подскуп информација који је изузет приликом реализације пројекта), чиме би се могле открити неконзистентности између претварања и стварног осећања одређене емоције, стим да се не решава проблем манипулативног понашања. Други смер унапређења захтева увођење бесконтактних сензора којима се могу детектовати биосигнали (на пример: термалном камером се може детектовати промена телесне температуре која је праћена емоционалном променом). На овај начин систем може осигурати детекцију манипулативног понашања.

Иако је у раду коришћено шест емоција, које представљају базичне емоције по Екману, постоји и треће унапређење које се може извести. Екман и други психолози дефинишу емоције у две групе: базичне и комплексне [4]. Рад се додатно може проширити коришћењем *Microsoft*-ове базе података која убраја чак 135 класа емоција на основу фацијалних експресија.

4. ЛИТЕРАТУРА

[1] Don H. Hockenbury, Sandra E. Hockenbury, “Discovering psychology”, *Worth Publishers*, април 2006.

[2] Steven J. Haggbloom, Renee Warnick, Jason E. Warnick, Vinessa K. Jones, Gary L. Yarbrough, Tenea M. Russell, Chris M. Borecky, Reagan McGahhey, John L. Powell III, Jamie Beavers, Emmanuelle Monte, “Review of General Psychology”, *American Psychology Association*, јун 2002.

[3] Manuela Angioni, Franco Tuveri, “Discovering Emotions through the Building of Linguistic Resource”, *Center for Research and Scientific Studies in Sardinia*, Italy, 2019.

[4] Robert Plutchnik, Henry Kellerman, “Theories of Emotion volume 1”, *Academic Press*, 1980.

Кратка биографија

Немања Томовић рођен је априла 1996. године у Новом Саду у Републици Србији. Одрастао је у Зрењанину, где је прве програмерске кораке начинио са једанаест година похађајући приватну школу за рачунарство и технику, а у Нови Сад се враћа због студирања. Завршио је смер мехатроника на основним академским студијама Факултета Техничких Наука дипломирајући из области компјутерске визије за детекцију коловозних линија са циљем аутоматске вожње. Годинама, током студија, али и након завршетка, бавио се израдом робота за Eurobot такмичење и био један од оснивача непрофитабилног удружења Invictus Robotics Association. Тренутно је запослен као софтверски архитекта за сигурност и безбедност возила.

Владимир Рајс рођен је 1982. године у Апатину. Дипломирао је 2007, а докторирао 2015. године на Факултету техничких наука у Новом Саду. Од 2016. године је био запослен као доцент, од 2021. као вандредни професор на Департману за електронику, енергетику и телекомуникације ФТН-а.

UDK: 621.38

DOI: https://doi.org/10.24867/31BE04Tomovic

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.31
DOI: https://doi.org/10.24867/31BE05Janosevic

ANALIZA SIGURNOSTI ELEKTROENERGETSKIH SISTEMA UZ PRIMENU KOREKTIVNIH AKCIJA

POWER SYSTEMS' SECURITY ANALYSIS AND REMEDIAL ACTIONS APPLICATION

Kristina Janošević, Luka Strezoski, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA i RAČUNARSTVO

Kratak sadržaj – *U radu su opisane teorijske osnove kriterijuma sigurnosti N-1 i korektivnih akcija koje se primenjuju u operativnom planiranju u elektroenergetskim sistemima. Predstavljen je opis jedne od procedura operativnog planiranja EES-a. Prikazani su rezultati dva primera primene korektivnih akcija nad mrežnim modelima u softverskom alatu Enterprise Transmission Network Analyzer 2.4 (eTNA).*

Ključne reči: *Elektroenergetski sistemi, Kriterijum sigurnosti N-1, Korektivne akcije, Operativno planiranje EES-a*

Abstract – *In the paper are described the theoretical basis of safety criteria N-1 and remedial actions applied in operational planning in power systems. A description of one of the operational planning procedures is presente.The results of two examples of applying remedial actions to the grid models in the software tool Enterprise Transmission Network Analyzer 2.4 (eTNA) are presented.*

Keywords: *Power systems, Security criterion N-1, Remedial actions, operational planning*

1.UVOD

Osnovna tri načela koja važe za svaki elektroenergetski sistem jesu pouzdanost, sigurnost i ekonomičnost. Sigurnost predstavlja sposobnost elektroeneretskog sistema da i dalje ostane u funkciji nakon svakog verovatnog poremećaja koji se može dogoditi, odnosno da se svi potrošači napajaju električnom energijom. Elektroenergetski sistem (EES) istovremeno mora biti funkcionalan tako da zadovoljava pomenuti uslov, ali i ekonomičan, otuda potreba da se nađe kompromis u sigurnosti tipa N-1, gde N označava broj komponenti sistema. Sistem je siguran tipa N-1 ukoliko posle kvara bilo koje, jedne od N komponenti, on i dalje ostane u normalnom pogonu [1]. Elektroenergetske mreže su prvobitno bile projektovane i konstruisane za ostrvski rad (jedna mreža – jedna država), nakon čega su izgrađeni interkonektivni dalekovodi prvenstveno zbog povezivanja EES-a i ostvarivanja, upravo, veće sigurnosti i pouzdanosti kao i lakše regulacije tako povezanog EES-a. Danas povezani EES-i omogućavaju veliki broj transakcija električne energije koje je teško u svakom trenutku fizički koordinisati i tehnički ispitati. Rad povezanog sistema se zasniva na principu da je svaki operator prenosnog sistema Rad povezanog sistema se zasniva na principu da je svaki operator prenosnog sistema (eng. *Transmission System Operator - TSO*) odgovoran za vlastitu mrežu, tj. svoju regulacionu oblast. Svaka kontrolna oblast i TSO su odgovorni za procedure kojima se obezbeđuje pouzdani rad sa aspekta rada u realnom vremenu, za ispade i havarijske situacije kao i pripremu iste. Kako bi se rizici što bolje procenili i sa njima se na adekvatan način suočili, a samim tim i osigurala efikasnost operativniih odluka i korektivniih akcija zahteva se bilateralna, multilaterna ili regionalna koordinacija između TSO-ova. Koordinacija se odvija i putem saradnje između regionalnog koordinatora sigurnosti (eng. *Regional Security Sentre - RSC*) i TSO-a gde je cilj obezbeđivanje sigurnosnih poboljšanja svih uključenih strana odgovornih za sigurnost EES-a.

U ovom radu je akcenat na objašnjenju osnovnog zadatka korektivnih akcija za siguran rad prenosnog sistema i da se razmotri kako bi stanje funkcionisalo kada bi se korektivne akcije uspešno primenjivale. Opisan je primer dve korektivne akcije koje su razmatrane u fazi operativnog planiranja u cilju poboljšanja situacije u mreži na spojenom modelu kontinentalne Evrope. Slučajevi koji su analizirani su rešenja preopterećenja na elemente od većeg značaja u mreži EES-a.

Kako bi se obezbedio siguran rad sistema, potrebno je da se obezbedi zaštita u slučaju pojava koje mogu prouzrokovati velike poremećaje u sistemu ili da iniciraju incidente većih razmera. Te pojave su kaskadni ispadi, propadi napona, odstupanje frekvencije od propisane vrednosti i gubitak sinhronizma. Pomenuti kriterijum N-1 primenjuju svi TSO-ovi i kombinuje se sa određenim izborom proizvodnih i prenosnih postrojenja i određivanjem dovoljne rezerve.

2. KRITERIJUM SIGURNOSTI N-1

U ovoj glavi su detaljno obrađeni prethodno pomenuti kriterijum sigurnosti N-1 kao najznačajniji za sprečavanje mogućih poremećaja u mreži. Kako bi se obezbedio siguran rad sistema, potrebno je da se obezbedi zaštita u slučaju pojava koje mogu prouzrokovati velike poremećaje u sistemu ili da iniciraju incidente većih razmera.

2.1 Osnovni pojmovi kod N-1 sigurnosti

Radno stanje unutar regulacione oblasti TSO-a, nakon bilo

NAPOMENA:

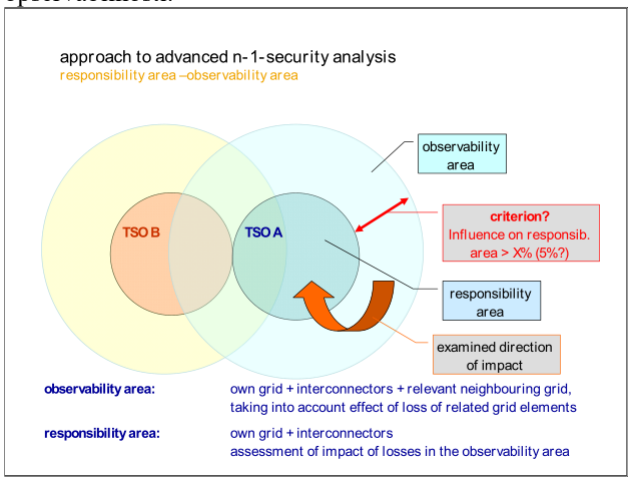
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Luka Strezoski.

kog pormećaja, ne sme izazvati gubitak značajnog dela potrošnje ili nekontrolisane kaskadne ispade koji se šire i van granica te oblasti ili se pak njihov uticaj oseća u susednim sistemima.

Poremećaj (eng. *Contingency*) predstavlja prekid, odnosno ispad jednog ili više elementa sa mreže koji se ne može unapred predvideti. Planirani ispadi (isključenja) se ne smatraju poremećajem, kao i dugoročni poremećaji koji se svrstavaju u planirane ispade. U analizi poremećaja (eng. *Contingency Analysis*) poremećaji se na osnovu verovatnoće nastanka dele na normalne (eng. *normal, ordinary*), izuzetne (eng. *exceptional*) i veoma retke (eng. *out-of-range*). Analiza poremećaja obuhvata sve elemente naponskog nivoa 400kV i više, ali se pored ovih razmatraju i elementi nižih naponskih nivoa (220kV, 150kV, 120kV, 110kV) ukoliko je njihov uticaj na siguran rad povezanog sistema značajan.

2.2 Definicija i analiza oblasti opservabilnosti

Definisanje oblasti opservabilnosti, u suštini, predstavlja osnov po kome TSO-ovi zahtevaju i dobijaju merenja određenih veličina u realnom vremenu iz prenosne mreže okolnih TSO-ova, a zatim se ta merenja implementiraju u SCADA sisteme nacionalnih dispečerskih centara. Svaki TSO ima dužnost da proverava uticaj okolnih sistema na svoju regulacionu oblast povremenim analizama. Kao rezultat toga dolaze pojmovi faktor uticaja, „prag“ uticaja poremećaja i lista eksternih poremećaja. Na taj način se obezbeđuju korektne simulacije uticaja prekograničnih ispada na nadležnu oblast. Svi eksterni elementi koji su od uticaja na regulacionu oblast TSO-a sa faktorom opservabilnosti većim od „praga“ opservabilnosti čine eksternu listu opservabilnosti. Oblast opservabilnosti sadrži regulacionu oblast TSO-a kao i eksternu mrežu, tako da je svaki TSO u stanju da konkretno simulira bilo koji poremećaj iz liste eksternih poremećaja tokom provere ispunjenosti kriterijuma sigurnosti N-1. Na slici 2.1 je definisana regulaciona oblast TSO-a i oblast opservabilnosti.



Slika 2.1 *Definicija oblasti opservabilnosti [3]*

3. KOREKTIVNE AKCIJE

Tokom puštanja N-1 simulacija postoji mogućnost da TSO-ovi detektuju preopterećenja na elementima svoje mreže. Tada je najvažnije rešiti potencijalne opasne situacije po mrežu i to na najefikasniji i najbrži način. U te svrhe se koristi pripremljen i proveren skup korektivnih akcija čiji je cilj da se u potpunosti zadovolji kriterijum sigurnosti N-1.

3.1 Definicija i osnovne podele

Korektivne akcije (eng. *Remedial actions*) se definišu za dva tipa ograničenja koja se mogu narušiti u mreži: ograničenje usled tokova snaga i naponsko ograničenje.

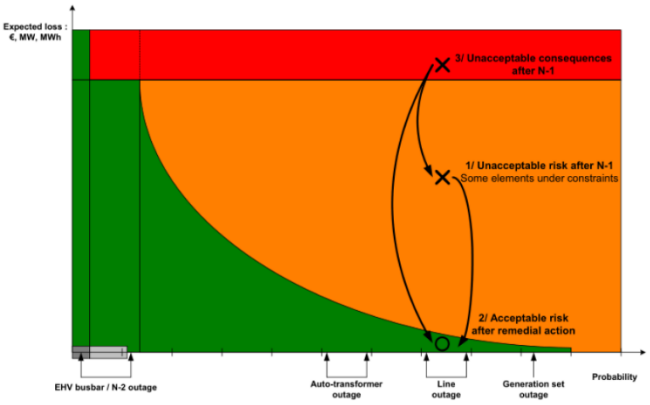
Preventivne akcije (eng. *Preventive remedial actions*) su mere koje se primenjuju pre nastanka poremećaja i imaju za cilj da preduhitre narušenje ograničenja koja usledila, a ne mogu biti ublažena na brz i efikasan način odmah nakon nastanka poremećaja.

Druge su kurativne akcije (eng. *Curative remedial actions*) – mere potrebne za suočavanje i brzo ublažavanje prekograničenih ograničenja, primenjuju se nakon nastanka poremećaja. Primenjene su sa zakašnjenjem postizanja potpune efikasnosti kompatibilne sa privremenim dozvoljenim opterećenjem (eng. *Temporary Admissible Transmission loading - TATL*). S obzirom na troškove koji su potrebni za njihovu aktivaciju dele se na jeftine i skupe.

S obzirom na troškove koji su potrebnu za njihovu aktivaciju dele se na jeftine i skupe.

3.2 Primena korektivnih akcija

Kako bi na vreme bio pripremljen, TSO u saradnji sa okolnim TSO-ovima ili sam treba unapred da pripremi moguće korektivne akcije koje bi se u slučaju poremećaja aktivirale. Ovo se obezbeđuje u fazi operativnog planiranja gde pored pripreme treba proveriti efikasnost akcija proračunom N-1 sigurnosti i tokova snaga, a primenjuju se u realnom vremenu. Bilo koje korektivne akcije koje imaju prekogranični uticaj moraju unapred biti dogovorene sa susednim TSO-ovima, takođe, akcije se moraju prilagoditi trenutnom stanju sistema i primeniti u koordinaciji sa okolnim TSO-ovima na koje utiču. Preventivne akcije se definišu i primenjuju pre nastanka poremećaja, dok se kurativne definišu unapred, a primenjuju nakon nastanka poremećaja. Ukoliko kurativne akcije ne postoje za neki tip poremećaja, u tom slučaju se primenjuju preventivne akcije. Na slici 3.1 su prikazane posledice primene korektivnih akcija na krivoj rizika. Vreme za sprovođenje korektivnih akcija je jedan od ključnih faktora, kako bi sistem funkcionisao unutar ograničenja koja su nametnuta tokovima snaga i naponskim prilikama. Regionalna koordinacija ima u ovome veliku ulogu jer se na taj način podstiče kontaktiranje TSO-ova u najkraćem mogućem roku i olakšava se pronalaženje mogućih rešenja problema.



Slika 3.1 *Posledice primene korektivnih akcija na krivoj*

rizika [4]

3.3 Mere koje se primenjuju kao korektivne akcije

Korektivne akcije mogu imati uticaj i na tržište pa se preporučuje da izbor korektivnih akcija bude usmeren tako da se ima najmanji uticaj na tržište. Obično se prvo primenjuju akcije koje se tiču promene topologije mreže. Pri pojavi ograničenja TSO mora prvo da razmatra promene u topologiji sopstvene mreže. Ukoliko se analizom sigurnosti ustanovi da promene u topologiji sopstvene mreže nisu dovoljne za siguran rad sistema, TSO poziva susede kako bi ustanovio kakve su mogućnosti za promenu njihove topologije, a koje će se pozitivno odraziti na ograničenja koja se javljaju u njegovoj mreži. nekim EES-ima u upotrebi su transformatori za promenu ugla (eng. *Phase Shift Transformers*). Predstavljaju posebnu vrstu interkonektivnih transformatora koji služe za upravljanje tokovima snaga po prenosnim i interkonektivnim dalekovodima. Još jedna od mogućih akcija koja se primenjuje je koordinisani redispečing za ublažavanje internih ograničenja ili onih na granicama. Prioritet je redispečing domaćih (TSO u kome se primenjuju akcije) proizvodnih jedinica.

4. ANALIZA KOREKTIVNIH AKCIJA

Praktični deo ovog rada je obuhvatio analize tokova snaga i N-1 kriterijum sigurnosti na pojednim spojenim modelima kontinentalne Evrope.

4.1 Korektivna akcija promena toka po MONITA kablu

Pri analizi sigurnosti N-1 na spojenom modelu kontinentalne Evrope za vremenski horizont dan unapred za datum 18.05.2023. su detektovana preopterećenja interkonektivnog dalekovoda između Crne Gore i Bosne i Hercegovine 220kV Trebinje – HE Perućica pri simuliranom ispadu interkonektivnog dalekovoda 400kV Trebinje – Lastva kao i preopterećenje transformatora prenosnog odnosa 400/220kV u TS Trebinju pri ispadu internog dalekovoda 400kV Trebinje – TE Gacko. Najkritičnija preopterećenja ova dva elementa prognozirana su za 18. maj u jutarnjem satu 06:30. Kritična preopterećenja elemenata se javljaju i za sate 11:30, 18:30 i 21:30. U baznom stanju (N stanje) opterećenje monitorisanih elementa za najkritičnije vremenske intervale je dato u tabeli 4.1.

Element	Vremenski interval	LOAD [%]
220kV Trebinje – HE Perućica	06:30	60.8
	11:30	44.9
	18:30	54.8
	21:30	57.3
TR 400/220kV Trebinje	06:30	78.8
	11:30	68.9
	18:30	74.5
	21:30	77.9

Tabela 4.1 *Opterećenje dalekovoda 220kV Trebinje – HE Perućica i transformatora 400/220kV u Trebinju u Base case stanju*

Nakon simulacije ispada dalekovoda 400 kV Trebinje – Lastva, odnosno u razmatranom stanju N-1, pri analizi sigurnosti vrednosti preopterećenja dalekovoda 220kV Trebinje – HE Perućica su data u tabeli 4.2.

Element	Vremenski interval	LOAD [%]
220kV Trebinje – HE Perućica	06:30	130.7
	11:30	114.3
	18:30	123.4
	21:30	126.8
TR 400/220kV Trebinje	06:30	131.1
	11:30	108.6
	18:30	125.6
	21:30	130.8

Tabela 4.2 *Preopterećenje dalekovoda 220kV Trebinje – HE Perućica i transformatora 400/220kV u Trebinju u N-1 stanju*

Prema informacijama iz razmena tokova između TSO-ova uočeno je da je tok aktivne snage za sve sate za 18.05. na MONITA kablu (HVDC kabl koji spaja Italiju i Crnu Goru [5]) bio planiran na maksimalnih, 600MW i to tako da je smer planiran ka Italiji.

Nakon dostavljenih rezultata analiza sigurnosti došlo se do zaključka da postoji opasnost od mogućih kaskadnih ispada nakon ovog preopterećenja. Kako bi došlo do rasterećenja dela mreže koji je opterećeniji gde spadaju i razmatrani elementi, predložena je korektivna akcija smanjenja toka aktivne snage po MONITA kablu, ovo je akcija koja zahteva koordinaciju više TSO-a. U ovom slučaju je razmatrano smanjenje aktivne snage na 300MW, odnosno 50% u odnosu na tokove koji su planirani. S obzirom da je ova akcija razmatrana u fazi operativnog planiranja, spada u grupu preventivnih korektivnih akcija.

Rezultati ponovljene analize sigurnosti analiziranih elemenata pri simularinim ispadima koji su identifikovani kao kritični su dati u tabeli 4.3.

Element	Vremenski interval	LOAD [%]
220kV Trebinje – HE Perućica	06:30	114.7
	11:30	95.7
	18:30	106.9
	21:30	110.9
TR 400/220kV Trebinje	06:30	108.3
	11:30	89.8
	18:30	102.2
	21:30	107.1

Tabela 4.3 *Preopterećenje dalekovoda 220kV Trebinje – HE Perućica i transformatora 400/220kV u Trebinju u N-1 stanju nakon primene korektivne akcije*

Upoređivajući rezultate iz tabela 4.3 i 4.2 po vremenskim intervalima može se primetiti da je za element dalekovod 220kV Trebinje – HE Perućica smanjeno preopterećenje za nekih 15 do 18%, dok je za transformator 400/220kV u Trebinju smanjeno preopterećenje 22-23% nakon modelovane korektivne akcije.

4.2 Korektivna akcija zatvaranje sklopnog uređaja spojnog polja (spajanje sabirnica) u TS 110kV Valjevo 3

Razmatrana je situacija, kao i u prethodnom primeru, u fazi operativnog planiranja, na spojenom mrežnom modelu kontinentalne Evrope za vremenski horizont dan unapred.

Analizirani su rezultati TSO-a Srbije za dan 21.10.2022. godine za sve sate. Jedno od kritičnijih preopterećenja koje je uzeto u razmatranje u ovom radu je element 110kV mreže Valjevo 3 – Valjevo 1. Razmatrane su situacije za dva slučaja: 1) u jednom delu dana (konkretno u ovom primeru samo vremenski interval 09:30) jedan od navedena dva transformatora je bio isključen, dok je spojno polje u uključeno i 2) oba transformatora 220/110 kV su uključena, ali je spojno polje u TS Valjevo 3 na 110kV tada isključeno. Upravo tada se javlja preopterećenje jednog od dalekovoda 110kV Valjevo 3 – Valjevo 1 i to za N-1 stanje sistema, odnosno za simulirani ispad jednog od pomenuta dva transformatora. Cilj korektivne akcije koja je diskutovana u ovom slučaju sa EMS-om je upravo povezivanje sabirnica na 110kV jer bi u tom slučaju bila povezana oba dalekovoda koja su inače u paraleli kao i oba transformatora i došlo bi do bolje raspodele snage. U tabeli 4.4 je prikazano stanje elemenata u baznom stanju. Vremenski interval 09:30 predstavlja slučaj kada je TR 1 220/110kV u Valjevu 3 isključen, a spojno polje u TS Valjevo 3 uključeno, a 12:30 predstavlja obrnut slučaj.

Element	Vremenski interval	LOAD [%]
110kV Valjevo 3 – Valjevo 1	09:30	103.4
	12:30	59.6
TR 2 220/110kV Valjevo 3	09:30	38.8
	12:30	45.4

Tabela 4.4 *Opterećenje 110kV Valjevo 3 – Valjevo 1 i transformatora 220/110kV u Valjevu 3 u Base case stanju*

Situacija u N-1 stanju koja se prva razmatra je u 09:30 preopterećenje TR 2 220/110kV Valjevo 3 pri simuliranim ispadima dalekovoda u oblasti Beograda, s obzirom da je drugi transformator koji je u paralelnom radu sa njim isključen, ova situacija je relevantna. Najveće preopterećenje je pri ispadu 110kV Beograd 3 – Beograd 16 i iznosi 120% (tabela 4.5).

Ispad elementa	Preopterećen element	Vremenski interval	LOAD [%]
110kV Beograd 3 – Beograd 16	TR 2 220/110kV Valjevo 3	09:30	120.1

Tabela 4.5 *Preopterećenje transformatora 220/110kV u Valjevu 3 u N-1 stanju*

Drugi slučaj u 12:30, kako je već rečeno spojno polje u TS 110kV Valjevo 3 je isključeno i TR 1 220/110kV je vraćen u pogon, preopterećuje se dalekovod 110kV Valjevo 3 – Valjevo 1 (tabela 4.6). Ovo je logično da se desi pri simulaciji s obzirom da se na istom sistemu sabirnica nalaze ova dva elementa.

Ispad elementa	Preopterećen element	Vremenski interval	LOAD [%]
TR 2 220/110kV Valjevo 3	110kV Valjevo 3 – Valjevo 1	12:30	104.5

Tabela 4.6 *Preopterećenje dalekovoda 110kV Valjevo 3 – Valjevo 1 u N-1 stanju*

Korektivna akcija koja bi trebala da popravi situaciju preopterećenja navedenog dalekovoda je spajanje sistema sabirnica, odrađena je analiza za 12:30 i utvrđeno da se preopterećenje više ne javlja, tj ispod 90% je.

5. ZAKLJUČAK

Jedan od osnovnih zadataka svakog TSO-a je da obezbedi siguran rad prenosnog sistema. Pojave koje mogu prouzrokovati velike poremećaje su kaskadni ispadi, propadi napona, odstupanje frekvencije od propisane vrednosti i gubitak sinhronizma. Zbog toga je svaki TSO dužan da u svom sistemu ima zadovoljen N-1 kriterijum sigurnosti.

S obzirom da postoji potreba za velikim tranzitom električne energije među susednim ili okolnim TSO-ovima, preopterećenja i zagušenja u mreži su postala sve učestalija pojava. U regionu jugoistočne Evrope korektivne akcije još uvek nisu zaživele na nivou koji je očekivan. Iako je to propisano, retko kada ima planiranja akcija unapred, već se sva zagušenja otklanjaju u realnom vremenu dispečerskim akcijama. Puštanje u pogon kabla između Italije i Crne Gore je bilo značajno za ovaj region ne samo jačanjem elektroenergetskog sistema Zapadnog Balkana već i što je pokrenulo ovu temu.

Generalno, može se reći da u regionu jugoistočne Evrope nije bilo velikih problema i potreba za detaljnijim planiranjem korektivnih akcija, međutim, svedoci smo da se situacija u poslednje vreme menja. Ono na čemu se i dalje mora raditi je pre svega podizanje svesti u regionu jugoistočne Evrope da će uskoro i kod nas biti još većeg udela u proizvodnji iz obnovljivih izvora te da će se menjati i način upravljanja sistemom.

6. LITERATURA

[1] Osnovni proračuni elektroenergetskih sistema, Tom 1, Vladimir Strezoski, Univerzitet u Novom Sadu, Fakultet tehničkih nauka

[3] ENTSO-E Continental Europe Operation Handbook - Policy 3: Operational Security, Final version, 2009.

[4] ENTSO-E Continental Europe Operation Handbook - Appendix 3: Operational Security, Final version, 2009.

[5] HVDC LINK CRNA GORA-ITALIJA ISPITIVANJE I PUŠTANJE U POGON, Zbornik radova, Treći dani elektro inženjera, IKCG, 24. - 25. OKTOBAR 2019

Kratka biografija

Kristina Janošević rođena je u Zaječaru 1997. god. Osnovne studije na Fakultetu tehničkih nauka na odseku elektroenergetski sistemi završila je 2020. godine
kontakt: kika.janosevic@gmail.com

dr Luka Strezoski rođen je u Novom Sadu 1990. Doktorirao je na Fakultetu tehničkih nauka 2017. god., a od 2020 je izabran za šefa katedre. Oblast interesovanja su elektroenergetski sistemi.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.9
DOI: https://doi.org/10.24867/31BE06Dobrokes

VEŠTAČKA INTELIGENCIJA I KOGNITIVNI SERVISI U AMAZONU

ARTIFICIAL INTELLIGENCE AND COGNITIVE SERVICES IN AMAZON

Branislav Dobrokes, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Veštačka inteligencija (AI) je oblast računarskih nauka posvećena rešavanju kognitivnih problema koji su obično povezani sa ljudskom inteligencijom, kao što su učenje, kreacija i prepoznavanje slika. Kognitivni servisi Amazona predstavljaju ključni segment inovacija u okviru Amazon Web Services (AWS), nudeći napredne alate zasnovane na veštačkoj inteligenciji koji mašinama omogućavaju da oponašaju ljudske sposobnosti percepcije, razumevanja i odlučivanja.

Ključne reči: *AI, AWS, Kognitivni servisi*

Abstract – Artificial intelligence (AI) is a field of computer science dedicated to solving cognitive problems typically associated with human intelligence, such as learning, creation, and image recognition. Amazon's cognitive services represent a key segment of innovation within Amazon Web Services (AWS), offering advanced tools based on artificial intelligence that enable machines to mimic human abilities of perception, understanding, and decision-making.

Keywords: *AI, AWS, Cognitive services*

1. UVOD

Veštačka inteligencija trenutno važi za jednu od najpopularnijih grana u tehnološkom sektoru. Cilj AI je stvaranje sistema koji su sposobni da sami uče i koji izvlače značenje iz podataka. Zatim, AI može primeniti to znanje da rešava nove probleme na načine slične ljudima.

Amazon Web Services (AWS) je platforma web usluga koju je Amazon pokrenuo 2006. godine, a ona pruža IT infrastrukturne usluge kompanijama kao web usluge, što je poznato kao cloud computing. Najistaknutije usluge koje AWS pruža su EC2, koji nudi virtuelne mašine, i S3, koji nudi kapacitet za skladištenje. Dostupno je preko 200 AWS cloud-based servisa širokog spektra u oblastima kao što su računarsvo, skladištenje, baze podataka, servisi za analitiku, mrežne usluge, mobilne usluge, alate za programere, IoT, sigurnost. Popularnost veštačke inteligencije (AI) i njena primena u industriji su drastično porasle u poslednje vreme, a AWS nudi širok spektar AI usluga sa izuzetno jednostavnom integracijom napredne inteligencije unutar aplikacija bez potrebe za iskustvom u oblasti mašinskog učenja (machine learning).

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Srđan Vukmirović, red. Prof.

2. VEŠTAČKA INTELIGENCIJA

Koreni veštačke inteligencije mogu se uočiti još tokom postojanja drevnih civilizacija gde su mitovi i priče često prikazivali inteligentna bića. Već tu se može primetiti ljudska težnja za stvaranjem veštačkih entiteta sa inteligencijom nalik ljudskoj. Međutim, termin veštačke inteligencije počinje formalno da se koristi tek sredinom 20. veka. Alan Turing, britanski matematičar, prvi je put upotrebio pojam veštačke inteligencije 1950. godine u radu "Računarska mašinerija i inteligencija". U savremenom dobu tehnološkog napretka, veštačka inteligencija izaziva značajnu pažnju kako u akademskim tako i u industrijskim krugovima. Sa širokim spektrom primena, od automatizacije do analize podataka, ona menja način na koji interagujemo sa digitalnim svetom.

2.1. Podela prema sposobnostima

Podela veštačke inteligencije prema sposobnostima obuhvata tri osnovne kategorije: usku veštačku inteligenciju, opštu veštačku inteligenciju i veštačku superinteligenciju. Ova klasifikacija se bazira na nivou sposobnosti i kompetencija koju sistemi veštačke inteligencije mogu da dostignu.

2.2. Podela prema funkcionalnostima

U okviru veštačke inteligencije, podela prema funkcionalnosti omogućava razumevanje različitih načina na koje AI sistemi obavljaju zadatke. Ova podela se fokusira na specifične funkcije koje AI sistemi mogu da ispune, i može se grupisati u četiri osnovne kategorije: reaktivne mašine, mašine sa ograničenom memorijom, teorija uma i samosvest.

2.3. Mašinsko učenje

Mašinsko učenje je ključna grana veštačke inteligencije (AI). Zasniva se na ideji da mašine mogu učiti iz podataka i na osnovu njih donositi zaključke. Sistemi zasnovani na mašinskom učenju efikasno koriste obimne baze podataka za unapređenje u različitim oblastima kao što su prepoznavanje lica, analiza govora, identifikacija objekata, prevođenje teksta i drugo. Ovo se razlikuje od tradicionalnog programiranja, gde se za svaki zadatak unapred definišu konkretna uputstva [3] .

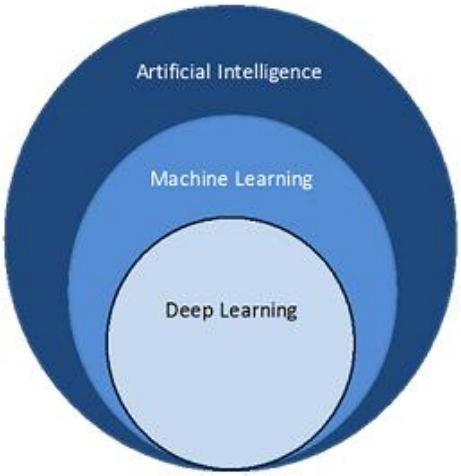
Mašinsko učenje omogućava sistemima da samostalno identifikuju obrasce i da na osnovu njih prave predikcije. Cilj mašinskog učenja je razvoj softvera koji može da automatski usvaja znanja iz prošlih iskustava i da svoje ponašanje u učenju postepeno unapređuje, omogućavajući mu da pravi predviđanja na osnovu novih podataka [3] .

Iako je mašinsko učenje deo veštačke inteligencije, često se dešava da se ovi termini koriste kao sinonimi. Mašinsko učenje predstavlja temelj na kojem veštačka inteligencija stoji, koristeći obimne skupove podataka za analizu i učenje. Kroz proces učenja iz podataka, mašinsko učenje omogućava sistemima da razumeju kontekst, interpretiraju informacije i donose odluke kada su uslovi neizvesni. Algoritmi mašinskog učenja igraju vitalnu ulogu u sistemima veštačke inteligencije, omogućavajući im da identifikuju trendove i prepoznaju obrasce u podacima.

2.4. Duboko učenje

Duboko učenje je napredna metoda mašinskog učenja koja se bazira na konceptu dubokih neuronskih mreža. Ova metoda predstavlja podskup mašinskog učenja, koristeći tehnike mašinskog učenja za rešavanje složenih problema putem neuronskih mreža koje imitiraju procese donošenja odluka kod ljudi. Duboko učenje omogućava mašinama da vrše zadatke na prirodan način, slično ljudskom mozgu, koristeći slojevitu strukturu za gradnju i proširenje znanja. Iako ovaj pristup zahteva značajne resurse i velike skupove podataka za obuku, on omogućava razvoj izuzetno sposobnih modela koji mogu da izvode složene zadatke sa velikom tačnošću [3] .

Korišćenjem dubokog učenja, razvijeni su sistemi koji mogu da prepoznaju govor, prepoznaju objekte u slikama, prevode jezike i izvode druge kompleksne zadatke koji zahtevaju napredno razumevanje i analizu. Ova metoda učenja se ističe sposobnošću da automatski identifikuje značajne karakteristike u velikim skupovima podataka, što značajno smanjuje potrebu za ručnim određivanjem karakteristika od strane programera. Jedan od izazova dubokog učenja je potreba za obimnim računarskim resursima i velikim količinama podataka za obuku, što može dovesti do značajnih troškova i izazova u obradi. Međutim, uprkos ovim izazovima, duboko učenje je dokazalo svoju vrednost kroz impresivne rezultate u raznim oblastima primene, postavši ključni element u razvoju naprednih veštačkih inteligencija koje mogu da obavljaju zadatke sa velikom preciznošću i efikasnošću.



Slika 1. Hijerarhija veštačke inteligencije

2.4. Primena veštačke inteligencije

Veštačka inteligencija ima širok spektar primena u različitim industrijama i oblastima. Sa svojom sposobnošću da obrađuje velike količine podataka, uči iz iskustava i automatski donosi odluke, veštačka inteligencija transformiše tradicionalne pristupe i otvara nove mogućnosti za rešavanje kompleksnih problema. Veštačka inteligencija se koristi u mnogim oblastima, uključujući obradu prirodnog jezika, analizu slika i videa, robotiku i automatizaciju, sisteme za preporuke, finansijske usluge, zdravstvo, virtuelne asistente i četbotove, igre, pametne kuće i IoT, kao i računarsku bezbednost.

3. KOGNITIVNI SERVISI U AMAZONU

Kognitivni servisi Amazona predstavljaju ključni segment inovacija u okviru Amazon Web Services (AWS), nudeći napredne alate zasnovane na veštačkoj inteligenciji koji mašinama omogućavaju da oponašaju ljudske sposobnosti percepcije, razumevanja i odlučivanja. Kognitivni servisi su tehnologije koje omogućavaju računarima da obrađuju informacije na sličan način kao ljudski mozak, uključujući razumevanje jezika, prepoznavanje objekata i zvukova, te učenje iz interakcija i podataka.

Amazon Web Services nudi obiman skup alata koji omogućavaju programerima da razviju sofisticirane i inteligentne aplikacije sposobne za interakciju s korisnicima na dubokom i intuitivnom nivou. Ovi alati su temelj za stvaranje rešenja koja otvaraju nove mogućnosti za inovacije, unapređujući kako komuniciramo i koristimo digitalne tehnologije u svakodnevnom životu, čime se oblikuje budućnost korišćenja kognitivnih servisa u digitalnom svetu.

3.1. Podela servisa

Ovi servisi se dele na četiri kategorije koje imitiraju ljudsku kogniciju: razumevanje jezika koje omogućava analiziranje i reagovanje na ljudski jezik, govor koji omogućava konverziju između govora i teksta, vizuelno prepoznavanje koje se koristi za identifikaciju objekata, lica i tekstova u slikama i videu, i prediktivna analitika koja koristi analizu podataka za donošenje odluka.

3.2. Kognitivni servisi za razumevanje prirodnog jezika

Kognitivni servisi za razumevanje prirodnog jezika predstavljaju jedan od najzanimljivijih aspekata današnje veštačke inteligencije, oni stvaraju vezu između ljudske komunikacije i obrade od strane mašina. Ovi servisi omogućavaju računarima ne samo da „čitaju“ i „razumeju“ tekst na način sličan ljudskom, već i da tumače kontekst i osećanja koja se prenose jezikom. To postižu koristeći napredne algoritme i modele mašinskog učenja, pored toga analiziraju podatke i prepoznaju ključne entitete, veze, teme i emocije, te na taj način otvaraju nove mogućnosti za automatizaciju, analizu i personalizaciju interakcija.

Primena ove tehnologije je široka, od unapređenja korisničkog iskustva u digitalnim asistentima, preko efikasne obrade i analize velikih skupova podataka, do razvoja sofisticiranih sistema koji mogu voditi smislene

dijaloge s korisnicima. Servisi za razumevanje prirodnog jezika su ključni za stvaranje interfejsa koji mogu prirodno komunicirati s ljudima, pružajući odgovore, preporuke i uvide zasnovane na razumevanju ljudskog jezika. Razvoj i implementacija ovih tehnologija predstavlja značajan napredak u približavanju mašina ljudskoj sposobnosti razumevanja i obrade jezika.

Amazon Lex predstavlja naprednu uslugu veštačke inteligencije koju nudi Amazon Web Services (AWS), dizajniranu sa ciljem omogućavanja razvoja interaktivnih interfejsa zasnovanih na razgovoru, kao što su četbotovi i virtuelni asistenti. Osnovna ideja iza Amazon Lex jeste da se programerima pruži alat koji kombinuje duboko učenje, razumevanje prirodnog jezika (NLU) i automatsko prepoznavanje govora (ASR) kako bi se stvorili razgovorni interfejsi koji mogu voditi prirodne dijaloge sa korisnicima. Ovo omogućava aplikacijama da postanu interaktivnije i pristupačnije, poboljšavajući korisničko iskustvo i omogućavajući sofisticiranu automatizaciju servisnih zadataka [4] .

Amazon Comprehend predstavlja kognitivni servis koji koristi obradu prirodnog jezika da izvuče uvide iz tekstualnih dokumenata. Ovaj servis kreira uvide prepoznavajući entitete, ključne fraze, jezike, sentimente i druge zajedničke elemente u dokumentu. Koristi se za kreiranje novih proizvoda zasnovanih na razumevanju strukture dokumenata. Na primer, upotrebom Amazon Comprehend mogu se pretraživati objave na društvenim mrežama za pominjanje proizvoda ili pregledati celokupni repozitorijum dokumenata za ključne fraze. Moguće je izvršiti analizu u realnom vremenu za manje obime podataka ili pokrenuti asinhrone zadatke analize za velike setove dokumenata. Korisnici mogu koristiti unapred obučene modele koje nudi Amazon Comprehend, ili mogu obučiti sopstvene prilagođene modele za klasifikaciju i prepoznavanje entiteta [5].

Amazon Transcribe je inovativna usluga AWS-a koja koristi napredne algoritme mašinskog učenja da bi omogućila preciznu transkripciju govora u tekst, pružajući korisnicima mogućnost da efikasno pretvore audio sadržaj u pisani oblik. Ova platforma je osmišljena da se nosi sa izazovima različitih akcenata, nivoa buke u pozadini i raznovrsnosti jezika, čime se ističe kao ključan alat u modernim tehnološkim rešenjima za obradu govora.

Amazon Translate predstavlja uslugu automatskog prevoda koju nudi Amazon Web Services, koristeći najsavremenije tehnologije mašinskog učenja da precizno prevede tekst između brojnih svetskih jezika. Ova usluga je specijalno dizajnirana da pomogne kompanijama i programerima u premošćavanju jezičkih barijera, omogućavajući laku integraciju prevoda u aplikacije, sajtove i poslovne procese. Amazon Translate čini globalnu komunikaciju dostupnijom nego ikada pre, podržavajući širok spektar aplikacija, od veb i mobilnih aplikacija, do složenih sistema za analizu podataka. Amazon Translate kontinuirano unapređuje svoje algoritme mašinskog učenja, što rezultira sve boljom tačnošću i prilagođavanjem specifičnim potrebama korisnika [6].

3.3. Kognitivni servisi za govor

Kognitivni servisi za govor predstavljaju suštinski element savremene veštačke inteligencije, omogućavajući mašini da proizvodi i razume ljudski govor na prirodan način. Ovi servisi koriste napredne algoritme mašinskog učenja i obradu prirodnog jezika kako bi se postigao visok stepen automatizacije i personalizacije. Takođe, oni značajno doprinose unapređenju korisničkog iskustva kroz interaktivne i intuitivne glasovne interfejse. Kognitivni servisi za govor primenjuju se u obrazovanju i zdravstvu za poboljšanje komunikacije i automatizaciju unosa podataka. U korporativnom sektoru, olakšavaju korisničku podršku kroz brze i personalizovane glasovne odgovore.

Amazon Polly je primer takvog servisa, to je AWS cloud servis koji pretvara tekst u govor nalik ljudskom. Ova usluga omogućava razvoj aplikacija koje povećavaju angažovanost korisnika i pristupačnost sadržaja. Amazon Polly podržava više jezika i nudi širok spektar glasova koji zvuče prirodno. Korišćenjem Polly servisa, mogu se kreirati aplikacije koje koriste govor na različitim lokacijama i omogućavaju izbor idealnog glasa za korisnike [7] .

3.4. Kognitivni servisi za vizuelna prepoznavanja

Kognitivni servisi za vizuelno prepoznavanje na Amazon Web Services (AWS) predstavljaju skup moćnih alata koji omogućavaju korisnicima da analiziraju i razumeju vizuelne podatke kao što su slike i video zapisi. Ovi servisi koriste napredne algoritme mašinskog učenja i veštačke inteligencije kako bi automatski detektovali, prepoznavali i klasifikovali objekte, lica, tekstove i scene unutar vizuelnih medija.

AWS nudi nekoliko ključnih servisa za vizuelno prepoznavanje, svaki sa specifičnim funkcionalnostima i primenama. Ovi servisi omogućavaju kompanijama iz različitih industrija da unaprede svoje operacije kroz automatizaciju i poboljšanu analitiku vizuelnih podataka. Bilo da se radi o sigurnosnim sistemima, medijskoj produkciji, e-commerce platformama ili industrijskoj automatizaciji, ovi alati pružaju neophodne resurse za brzu i preciznu obradu velikih količina vizuelnih informacija.

Amazon Rekognition, cloud usluga za analizu slika i video zapisa koja omogućava lako dodavanje naprednih mogućnosti računarskog vida vašim aplikacijama. Servis koristi proverenu tehnologiju dubokog učenja i ne zahteva znanje iz mašinskog učenja za korišćenje. Amazon Rekognition uključuje jednostavan API koji brzo analizira bilo koju sliku ili video datoteku sačuvanu u Amazon S3 [8] .

Amazon Textract je napredan AWS servis koji omogućava automatsku detekciju i analizu teksta iz različitih dokumenata. Ovaj alat koristi napredne tehnike mašinskog učenja kako bi prepoznao i izvukao štampani i ručno pisan tekst iz finansijskih izveštaja, medicinskih zapisa, poreskih obrazaca i drugih dokumenata [9]. Amazon Textract pruža mogućnost ekstrakcije teksta, formi i tabela iz dokumenata pomoću Document Analysis API-a. U okviru Analyze Document API-a, korisnici

mogu specificirati i ekstraktovati informacije po želji. Ovaj alat podržava i obradu faktura i računa kroz AnalyzeExpense API, kao i identifikacionih dokumenata poput vozačkih dozvola i pasoša uz AnalyzeID API.

Amazon Lookout for Vision je AWS servis koji omogućava pronalaženje vizuelnih defekata u industrijskim proizvodima precizno i u velikom obimu. Koristeći računarski vid, ovaj servis može identifikovati nedostajuće komponente u industrijskim proizvodima, oštećenja na vozilima ili strukturama, nepravilnosti na proizvodnim linijama, pa čak i najsitnije defekte na silikonskim pločicama – ili bilo koji drugi fizički predmet gde je kvalitet važan, kao što je nedostajući kondenzator na štampanim pločama [10].

3.5. Kognitivni servisi za prediktivnu analitiku i mašinsko učenje

Kognitivni servisi za prediktivnu analitiku i mašinsko učenje na AWS-u predstavljaju snažne alate koji omogućavaju organizacijama da izvuku vredne uvide iz svojih podataka i predvide buduće trendove. Ovi servisi omogućavaju automatizaciju procesa donošenja odluka na osnovu podataka, što organizacijama pruža konkurentsku prednost u brzom i dinamičnom okruženju. AWS nudi platformu koja kombinuje skalabilnost, sigurnost i inovacije, omogućavajući korisnicima da brzo razvijaju, treniraju i implementiraju modele mašinskog učenja, čak i ako nemaju duboko tehničko znanje.Najpoznatiji servisi u ovoj oblasti su Amazon SageMaker, Amazon Forecast i Amazon Personalize, koji omogućavaju predviđanje poslovnih ishoda, personalizaciju korisničkih iskustava i predikciju na osnovu istorijskih podataka.

Amazon SageMaker je potpuno upravljani servis za mašinsko učenje (ML) koji omogućava programerima da brzo i sigurno grade, treniraju i implementiraju modele mašinskog učenja u produkciono spremno okruženje. SageMaker obezbeđuje korisnički interfejs za upravljanje radnim tokovima mašinskog učenja, čineći alate za ML dostupnim u više integrisanih razvojnih okruženja. Sa SageMaker-om možete skladištiti i deliti svoje podatke bez potrebe za izgradnjom i upravljanjem sopstvenim serverima, što omogućava vašoj organizaciji više vremena za kolaborativni razvoj ML radnih tokova i njihovu bržu implementaciju [11] .

Amazon Forecast je potpuno upravljan servis koji koristi napredne statističke metode i algoritme mašinskog učenja za precizno predviđanje vremenskih serija. Ovaj servis je zasnovan na istoj tehnologiji koju Amazon.com koristi za svoje prognoze i omogućava korisnicima da predviđaju buduće vrednosti na osnovu istorijskih podataka, bez potrebe za dubokim znanjem iz oblasti mašinskog učenja. Prognoze vremenskih serija su izuzetno korisne u mnogim oblastima, kao što su maloprodaja, finansije, logistika i zdravstvo. Uz pomoć Amazon Forecast-a, kompanije mogu preciznije planirati zalihe, radnu snagu, veb saobraćaj, kapacitet servera i finansijske tokove. Na primer, u maloprodaji ovaj servis može pomoći u predviđanju potražnje za proizvodima, što omogućava bolje upravljanje zalihama i cenama u različitim prodavnicama [12]. Korišćenjem Amazon Forecast-a, maloprodajne kompanije mogu smanjiti prekomerne zalihe, optimizovati nabavku i poboljšati dostupnost proizvoda, čime se povećava efikasnost poslovanja i zadovoljstvo kupaca.

4. ZAKLJUČAK

Veštačka inteligencija (AI) predstavlja ključnu tehnologiju 21. veka, sa potencijalom da transformiše sve aspekte našeg života, od poslovanja i zdravstva do obrazovanja i zabave. Njena sposobnost da obrađuje velike količine podataka, uči iz iskustava i donosi odluke na osnovu kompleksnih analiza čini je neophodnim alatom za rešavanje izazova savremenog doba. AWS pruža širok spektar usluga koje omogućavaju organizacijama da iskoriste prednosti veštačke inteligencije u svojim poslovnim procesima. Ove usluge pokrivaju različite oblasti, uključujući obradu prirodnog jezika, prepoznavanje govora, analizu slika i videa, kao i prediktivnu analitiku. Međutim, iako AWS nudi brojne prednosti, postoje i određeni izazovi koji prate primenu veštačke inteligencije. Jedan od njih je etička upotreba AI tehnologija, posebno u kontekstu privatnosti podataka i sigurnosti. Kako AI postaje sve više integrisan u naše živote, važno je osigurati da njena upotreba bude u skladu sa etičkim standardima i da ne dovede do negativnih posledica po pojedince i društvo u celini. Budućnost veštačke inteligencije izgleda izuzetno obećavajuće, sa potencijalom da duboko utiče na sve aspekte našeg društva. Kako AI tehnologije nastavljaju da se razvijaju, mogućnosti za njihovu primenu će se sve više širiti, otvarajući nove puteve za inovacije i razvoj.

5. LITERATURA

[1] https://www.britannica.com/topic/Amazoncom
[2] https://aws.amazon.com/what-is/artificial-intelligence/
[3] https://viso.ai/deep-learning/deep-learning-vs-machine-learning/
[4] https://docs.aws.amazon.com/lex/
[5] https://docs.aws.amazon.com/comprehend/
[6] https://docs.aws.amazon.com/translate/
[7] https://docs.aws.amazon.com/polly/
[8] https://docs.aws.amazon.com/rekognition/
[9] https://docs.aws.amazon.com/textract/
[10] https://docs.aws.amazon.com/lookout-for-vision/
[11] https://docs.aws.amazon.com/sagemaker/
[12] https://docs.aws.amazon.com/forecast/

Kratka biografija:

Branislav Dobrokes rođen je u Osijeku 1998. god. Studijski program Primenjeno softversko inženjerstvo upisao je 2017. godine, a završio 2021. godine. Nakon toga upisuje master akademske studije Primenjene računarske nauke i informatika, iz oblasti Elektrotehnike i računarstva.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.313
DOI: https://doi.org/10.24867/31BE07Vranjes

DINAMIČKI MODEL ŠESTOFAZNE KAVEZNE ASINHRONE MAŠINE

DYNAMIC MODEL OF A SIX-PHASE SQUIRREL-CAGE INDUCTION MACHINE

Milan Vranješ, Dejan Jerkan, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je prikazan detaljan dinamički model asimetrične šestofazne kavezne asinhrone mašine koji uzima u obzir više prostorne harmonike polja, tj. više harmonike induktivnosti namotaja. Parametri modela su određeni pomoću FEM modela mašine. Primenom formiranog modela analizirani su karakteristični prelazni procesi i ustaljeni radni režimi sa posebnim akcentom na prednostima složenijeg modela u odnosu na uobičajene modele koji u obzir uzimaju samo osnovne prostorne harmonike polja.*

Ključne reči: *Asimetrična šestofazna asinhrona mašina, FEM model, prelazni procesi*

Abstract *The paper presents a detailed dynamic model of an asymmetrical six-phase squirrel-cage induction machine that takes into account higher spatial harmonics of the field, i.e. higher harmonics of inductance of the coil. The model parameters are determined using the FEM model of the machine. By applying the formed model the characteristic transitional procceses and working steady regimes are analysed with particular emphasis on the advantages of a more complex model over convention models that take into account only the fundemental spatial harmonics of the field.*

Keywords: *Asymmetrical six-phase induction machine, FEM model, transitional processes*

1. Uvod

Poslednjih godina, višefazne asinhrone mašine sa kaveznim rotorom (AM), kao pogonska rešenja pokazuju veliki potencijal za implementaciju u električnim vozilima. Asimetrični šestofazni AM je jedan od najpopularnijih izbora za takve pogone, posebno jer se njegovi namotaji statora sastoje od dva trofazna sistema koji su nezavisno povezani u dve sprege Y. Da bi se istražilo ponašanje ovakvih višefaznih sistema, potrebno je raspolagati sa adekvatnim matematičkim modelom mašine. U ovom radu biće korišćena MCCA (Multiple Coupled Circuit Approach) metoda za matematičko modelovanje mašine, pri čemu se akvizicija matrice induktivnosti mašine vršiti serijom magnetostatičkih 2D FEA simulacija.

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

Magnetostatičke 2D-FEA simulacije su sa stanovišta numeričke složenosti najjednostavnije simulacije, koje se mogu izvesti sa zadovoljavajućom brzinom, pogotovo kada je pod određenim uslovima dopušteno mašinu modelovati magnetski linearnim materijalima. Prostorni oblici induktivnosti mašine dobijeni pomoću FEA simulacija se primenom MCCA metode koriste kao ulazni parametri u modelu mašine. Od načina njihove numeričke interpretacije zavisi i kvalitet razvijenog MCCA modela. Pored jednostavnije primene *lookup* tabela, postoji i rešenje u kojem se prostorni oblici induktivnosti razvijaju u Furijeov red i tako ugrađuju u MCCA model. U ovom radu je primenjeno upravo rešenje sa razvojem u Furijeov red, koje nudi mogućnost za uvažavanje ili odbacivanje pojedinih članova reda, kako bi se mogao selektivno istraživati njihov uticaj, što kod primene *lookup* tabela nije slučaj. Sa druge strane, *lookup* tabele su jednostavnije za implementaciju i manje su numerički zahtevne. Predloženi model mašine će biti verifikovan simulacijom test rutine (zalet mašine i nakon nekog vremena opterećivanje) i analizom karakterističnih električnih i mehaničkih veličina. Za potrebe ovog rada analizirana je šestofazna asimetrična AM, nazivne snage 1,1 kW. Mašina ima 36 žlebova na statoru i 33 na rotoru.

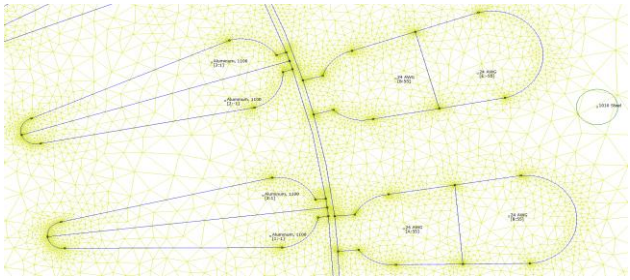
2. Implementacija dinamičkog modela asimetrične šestofazne asinhrone mašine

Kao što je istaknuto u prethodnom poglavlju, za modelovanje šestofazne asinhrone mašine biće korišćen dinamički MCCA model mašine, a akvizicija matrice induktivnosti izvršena primenom FEA simulacija. Dinamički MCCA model će biti implementiran u okruženju MATLAB&*Simulink*, a FEA simulacije će biti izvršene pomoću FEMM 4.2 alata. Integracija MATLAB&*Simulink* i FEMM 4.2 je omogućena primenom MATLAB-ovog *toolbox*-a OctaveFEMM [5].

2.1. Opis FEM modela i postupka za određivanje matrice induktivnosti mašine

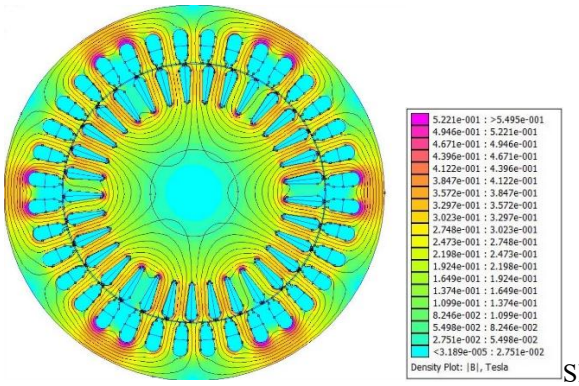
Za potrebe ovog rada je korišćen besplatni softver FEMM 4.2, koji je dostupan za preuzimanje sa stranice [4]. FEMM 4.2 omogućava uvoz geometrije problema iz drugih programskih paketa. Nakon ubacivanja geometrije se može pristupiti postupku definisanja svojstava materijala svakom disjunktnom delu crteža. Nakon definisanja materijala od kojeg je sačinjen određeni deo poprečnog preseka mašine, može se pristupiti i formiranju namotaja mašine, tako što se domenu koji pripada određenom statorskom žlebu dodeli i pripadnost određenom električnom kolu, sa odgovarajućim brojem provodnika koji su smešteni u taj žleb, a ujedno se zadaje i orijentacija samih provodnika. Program dozvoljava isključivo pobuđivanje strujnim izvorima, što predstavlja određenu

poteškoću koja se može prevazići kombinovanjem simulacija sa eksperimentalnim rezultatima. Nakon pravilno formulisane geometrije problema i svojstava svakog dela poprečnog preseka mašine, može se pristupiti definisanju gustine mreže konačnih elemenata. Detalj formirane mreže u okolini vazdušnog zazora – slika 2.1.



Slika 2.1 *Detalj mreže konačnih elemenata.*

Za potrebe izračunavanja induktivnosti je najpodesnije koristiti magnetostatičke simulacije u kojima se pobuđuje određeni namotaj i to jednosmernom strujom, pa se na osnovu izračunatih fluksnih obuhvata izračunavaju sopstvene i međusobne induktivnosti. Zbog zavisnosti induktivnosti od trenutnog položaja rotora potrebno je uraditi veći broj simulacija za različite položaje rotora.
U radu su izvršene simulacije sukvesivnom rotacijom rotora sa korakom od 1,5 mehaničkih stepeni, što znači ukupno 240 magnetostatičkih simulacija za jedan pun obrtaj. Na slici 2.2 je prikazan odziv magnetostatičke simulacije nad linearnim modelom mašine, za slučaj pobuđivanja namotaja faze a jednosmernom strujom intenziteta I=1 A. Primenom opisanog postupka dobijeni su diskretni nizovi svih induktivnosti u mašini u zavisnosti od položaja rotora.



Slika 2.2 *Odziv linearne magnetostatičke simulacije. Namotaj faze a pobuđen jednosmernom strujom intenziteta I=1 A.*

2.2. Razvijanje odbiraka induktivnosti u diskretan Furijeov red

Dobijene diskretne nizove svih induktivnosti u mašini u zavisnosti od položaja rotora je sada potrebno na adekvatan način interpretirati za potrebe dinamičkog modela. U ovom modelu je korišćen pristup koji se zasniva na činjenici da su sve induktivnosti u rotacionoj mašini periodične funkcije položaja rotora, te se stoga odbirci dobijeni diskretnim magnetostatičkim simulacijama mogu predstaviti preko koeficijenata Furijevog razvoja u red. Neka se razmatra periodična funkcija $f(x)$ sa osnovnom periodom 2π. Takva funkcija se može predstaviti preko sume u obliku Furijeovog reda (2.1), dok je definicija koeficijenata reda data relacijom (2.2).

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cdot \cos(nx) + \sum_{n=1}^{\infty} b_n \cdot \sin(nx) \tag{2.1}$$

$$a_0 = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)dx\,, a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\cdot\cos(nx)dx\,,$$
$$b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\cdot\sin(nx)dx \tag{2.2}$$

Rezultati dobijeni magnetostatičkim simulacijama su diskretne veličine, te se na njih zapravo primenjuje diskretizovana verzija razvoja u Furijeov red. Prikazanim postupkom se u potpunosti rekonstruiše matrica induktivnosti mašine, na način koji je izuzetno praktičan za računarsku implementaciju dinamičkog modela.

2.4. Opis MATLAB&Simulink MCCA dinamičkog modela

MCCA model mašine se zasniva na jednačinama naponske raznoteže. Električna kola (konture) u električnoj mašini su međusobno magnetno spregnute i zbog varijacije njihovog međusobnog položaja matrica induktivnosti je složena funkcija geometrije mašine i položaja rotora. Matrična jednačina (2.3) obuhvata sva električna kola u asinhronoj mašini, i faze statora i rotorske petlje (broj petlji rotora je jednak broju rotorskih štapova N_B, a jednačinama (2.4)-(2.12) su definisani svi vektori i submatrice koje se koriste u ovoj jednačini. Matrice induktivnosti stator-stator i rotor-rotor $\boldsymbol{L_{ss}}$ i $\boldsymbol{L_{rr}}$ mogu se smatrati konstantnim ako njihova mala varijacija kao rezultat ožljebljenosti nije od interesa, dok su matrice stator-rotor i rotor-stator $\boldsymbol{L_{sr}}$ i $\boldsymbol{L_{rs}}$ uvek periodične funkcije električnog ugla položaja rotora ϑ.

$$\begin{bmatrix}\boldsymbol{U_s}\\ \boldsymbol{U_r}\end{bmatrix} = \left(\begin{bmatrix}\boldsymbol{R_{ss}} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{R_{rr}}\end{bmatrix} + \frac{d}{dt}\begin{bmatrix}\boldsymbol{L_{ss}} & \boldsymbol{L_{sr}}\\ \boldsymbol{L_{rs}} & \boldsymbol{L_{rr}}\end{bmatrix}\right)\begin{bmatrix}\boldsymbol{I_s}\\ \boldsymbol{I_r}\end{bmatrix}, \tag{2.3}$$

$$\boldsymbol{U_s} = [U_a \quad U_b \quad U_c \quad U_d \quad U_e \quad U_f]^{\boldsymbol{T}}, \tag{2.4}$$

$$\boldsymbol{U_r} = [0 \quad \dots \quad 0]^T_{N_B}, \tag{2.5}$$

$$\boldsymbol{I_s} = [I_a \quad I_b \quad I_c \quad I_d \quad I_e \quad I_f]^{\boldsymbol{T}}, \tag{2.6}$$

$$\boldsymbol{I_r} = [I_{r1} \quad \dots \quad I_{rN_B}]^T_{N_B}, \tag{2.7}$$

$$\boldsymbol{R_{ss}} = diag(R_s \quad R_s \quad R_s \quad R_s \quad R_s \quad R_s), \tag{2.8}$$

$$\boldsymbol{R_{rr}} = diag(R_r \quad \dots \quad R_r), \tag{2.9}$$

$$\boldsymbol{L_{ss}} = \begin{bmatrix} L_a & M_{ab} & M_{ac} & M_{ad} & M_{ae} & M_{af}\\ M_{ba} & L_b & M_{bc} & M_{bd} & M_{be} & M_{bf}\\ M_{ca} & M_{cb} & L_c & M_{cd} & M_{ce} & M_{cf}\\ M_{da} & M_{db} & M_{dc} & L_d & M_{de} & M_{df}\\ M_{ea} & M_{eb} & M_{ec} & M_{ed} & L_e & M_{ef}\\ M_{fa} & M_{fb} & M_{fc} & M_{fd} & M_{fe} & L_f \end{bmatrix}, \tag{2.10}$$

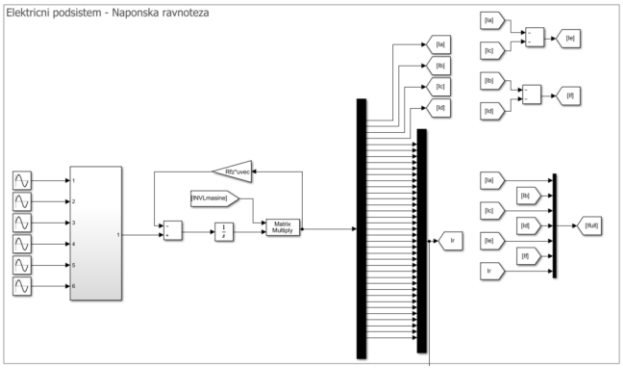
$$\boldsymbol{L_{sr}} = \begin{bmatrix} M_{ar_1} & M_{ar_2} & \dots & M_{ar_{N_B-1}} & M_{ar_{N_B}}\\ M_{br_1} & M_{br_2} & \dots & M_{br_{N_B-1}} & M_{br_{N_B}}\\ M_{cr_1} & M_{cr_2} & \dots & M_{cr_{N_B-1}} & M_{cr_{N_B}}\\ M_{dr_1} & M_{dr_2} & \dots & M_{dr_{N_B-1}} & M_{dr_{N_B}}\\ M_{er_1} & M_{er_2} & \dots & M_{er_{N_B-1}} & M_{er_{N_B}}\\ M_{dr_1} & M_{fr_2} & \dots & M_{fr_{N_B-1}} & M_{fr_{N_B}} \end{bmatrix}_{6xN_B}, \tag{2.11}$$

$$\boldsymbol{L_{rr}} = \begin{bmatrix} L_{r_1} & M_{r_{12}} & \dots & M_{r_{1(N_B-1)}} & M_{r_{1N_B}} \\ M_{r_{21}} & L_{r_2} & \dots & M_{r_{2(N_B-1)}} & M_{r_{2N_B}} \\ \vdots & \vdots & \dots & \vdots & \vdots \\ \vdots & \vdots & \dots & \vdots & \vdots \\ M_{r_{(N_B-1)1}} & 0 & \dots & L_{r_{N_B-1}} & M_{r_{(N_B-1)N_B}} \\ M_{r_{N_B1}} & M_{r_{N_B2}} & \dots & M_{r_{N_B(N_B-1)}} & L_{r_{N_B}} \end{bmatrix}_{N_B x N_B} \quad (2.12)$$

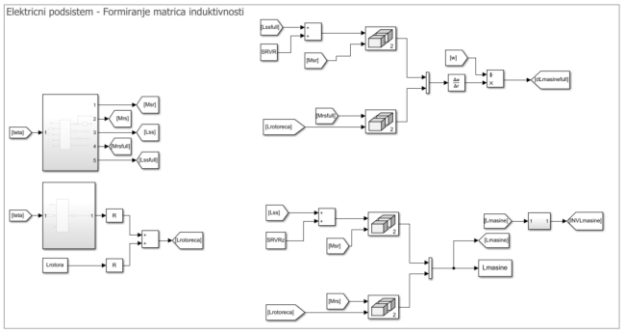
Generisanje obrtnog momenta u MCCA modelu asinhrone mašine je modelovano po principu virtuelnog rada. Sve submatrice su u opštem slučaju zavisne od ugla ϑ, tako da sve one učestvuju u stvaranju obrtnog momenta.

$$T_e = \frac{1}{2} p \begin{bmatrix} \boldsymbol{I_s} \\ \boldsymbol{I_r} \end{bmatrix}^T \frac{d}{d\vartheta} \begin{bmatrix} \boldsymbol{L_{ss}} & \boldsymbol{L_{sr}} \\ \boldsymbol{L_{rs}} & \boldsymbol{L_{rr}} \end{bmatrix} \begin{bmatrix} \boldsymbol{I_s} \\ \boldsymbol{I_r} \end{bmatrix} \quad (2.13)$$

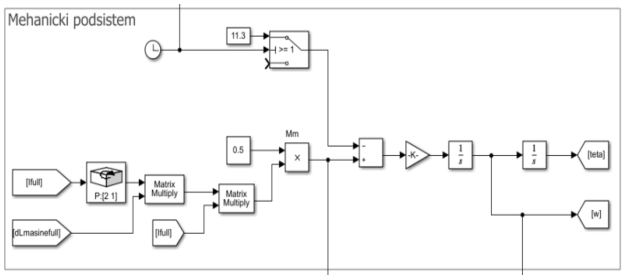
MCCA dinamički model mašine je implementiran u MATLAB&*Simulink* okruženju. Na slikama 2.3-2.5 su istaknuti karakteristični delovi *Simulink* modela.



Slika 2.3 *Električni podsistem modela (jednačine naponske ravnoteže).*



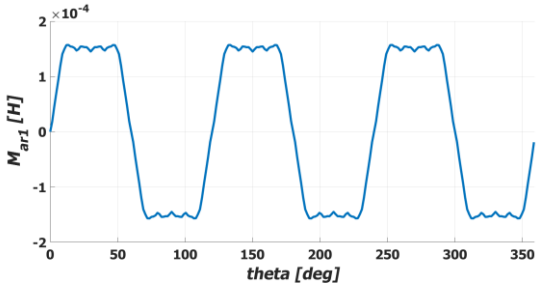
Slika 2.4 *Blokovi za dinamičko formiranje matrica induktivnosti.*



Slika 2.5 *Mehanički podsistem (Njutnova jednačina kretanja).*

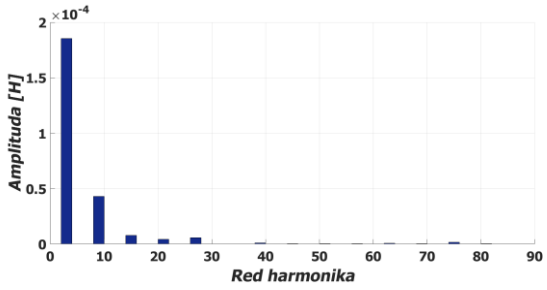
3. REZULTATI I DISKUSIJA REZULTATA SIMULACIJA

Na slici 3.1 prikazan je prostorni oblik međusobne statorsko-rotorske induktivnosti $M_{ar1}(\vartheta)$. Ona predstavlja međuinduktivnost statorske faze A i rotorske petlje 1. Prema očekivanjima, ova međuinduktivnost predstavlja pravu naizmeničnu veličinu jer je jedan od namotaja (rotorska petlja) u relativnom kretanju u odnosu na stojeći namotaj faze A. Karakterističan trapezasti oblik kojim je dominantno određena ova induktivnost jeste posledica raspodeljenosti statorskih namotaja kojima se postiže da magnetopobudna sila statora gledano sa strane rotora što više nalikuje prostoperiodičnoj. Uz trapezasti oblik se primećuju i manje oscilacije višeg harmonijskog reda, koje su direktna posledica postojanja žlebova na statoru.



Slika 3.1 *Međusobna statorsko-rotorska induktivnost $M_{ar1}(\vartheta)$; linearna magnetostatička simulacija.*

Trapezni oblik poseduje karakteristične harmonike neparnog reda (1, 3, 5, 7...). Kako je ovde izvršen razvoj talasnog oblika u punom mehaničkom obrtaju, pobrojani neparni harmonici će biti tri puta većeg reda (3, 9, 15, 21...) što se i jasno vidi na slici 3.2.

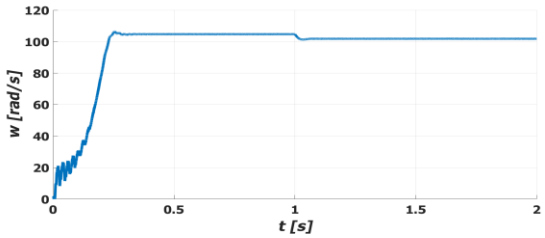


Slika 3.2 *Spektar viših harmonika međusobne induktivnosti $M_{ar1}(\vartheta)$.*

3.1. Prikaz i analiza odziva modela za izabrani test scenario – zaletanje mašine do ustaljenog stanja praćeno naglim povećanjem opterećenja

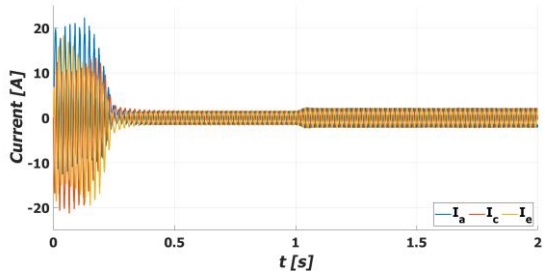
Nakon diskutovanih talasnih oblika induktivnosti će se pristupiti diskusiji odziva dobijenih simulacijama nad MCCA modelom u *Simulink*-u. Za testiranje modela je izabran sledeći profil prelaznog procesa: najpre se vrši zalet neopterećene mašine nametanjem punog napona napajanja (direktan start); Nakon uspešno obavljenog zaleta u trenutku t=1s mašini se skokovito nameće nominalni momenat opterećenja u iznosu od 11,3 Nm; Profil prelaznog procesa se okončava ustaljivanjem brzine na nižu vrednost određenu nametnutim opterećenjem, te se taj stacionarni režim dopušta do vremenskog trenutka t=2s

koji je određen za završetak simulacije. Na slici 3.3 je prikazan talasni oblik mehaničke brzine obrtanja nametnutog profila. Primećuje se prelazni proces tokom zaletanja mašine sa značajnije izraženim oscilacijama od onih karakterističnih za odzive klasičnih dinamičkih modela. Složenoperiodične induktivnosti mašine značajnije doprinose izobličenju elektromagnetnog momenta (pojava tzv. parazitinih momenata konverzije), što se podsredstvom Njutnove jednačine kretanja odražava i na talasni oblik uspostavljene brzine.



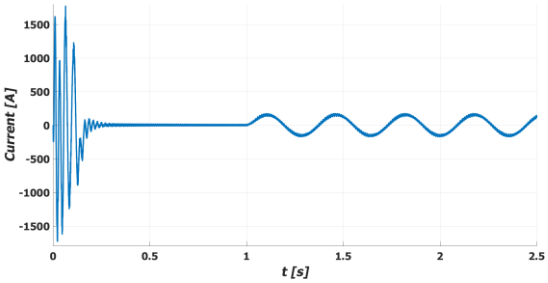
Slika 3.3 *Odziv mehaničke brzine mašine.*

Na slici 3.4 prikazani su talasni oblici faznih struja jednog od statorskih trofaznih namotaja (ACE). Primećuje se karakterističan porast amplitude tokom zaleta, kao i porast u odnosu na struje pri prelasku iz praznog hoda u nazivno opterecnje.



Slika 3.4 *Talasni oblici statorskih struja* I_a, I_c, I_e

Interesantno je prikazati i talasni oblik struje jedne od rotorskih petlji tokom razmatranog prelaznog procesa – slika 3.5 Primećuju se vrlo velike amplitude struje u početnim trenucima zaleta, te njihov pad na praktično nultu vrednost nakon postizanja ustaljenog režima praznog hoda (režim sa klizanjem kojem teži nuli – nema indukovanja struja). Nakon opterećivanja mašine nazivnim momentom dolazi do indukovanja struja u rotorskim petljama na frekvenciji koja je proporcionalna klizanju (7% od statorske učestanosti od 50 Hz) što se jasno vidi u sporopromenljivom talasnom obliku nakon trenutka t = 1s. Takodje se primećuje da je reč o složenoperiodičnom talasnom obliku u ustaljenom nazivnom režimu.



Slika 3.5 *Talasni oblik struje jedne rotorske petlje.*

3. ZAKLJUČAK

U radu je prikazan postupak određivanja diskretnih talasnih oblika induktivnosti šestofazne asimetrične asinhrone mašine primenom metode konačnih elemenata, pod pretpostavkom linearnosti magnetne sredine. Talasni oblici se zatim koriste u razvijanju dinamičkog MCCA modela, tako što su induktivnosti u matricama modela predstavljene preko koeficijenata razvoja u Furijeov red. Pokazano je da MCCA model efikasno uvažava pojavu viših harmonika u odzivima, a koji su posledica uvaženih složenoperiodičnih talasnih oblika induktivnosti. Model u sadašnjoj formi se može koristiti za istraživanje uticaja pojedinih vrsta kvarova u mašini na harmonijski sastav njenih terminalnih veličina (struja, momenta, brzine...), čime se otvara prostor za razvijanje neinvazivnih tehnika za otkrivanje unutrašnjih kvarova u mašinama.

4. LITERATURA

[1] D. G. Jerkan, D. D. Reljić and D. P. Marčetić, "Broken Rotor Bar Fault Detection of IM Based on the Counter-Current Braking Method," in IEEE Trans. on Energy Conversion, vol. 32, no. 4, pp. 1356-1366, Dec. 2017.

[2] M. Ojaghi, M. Sabouri and J. Faiz, "Performance Analysis of SquirrelCage Induction Motors Under Broken Rotor Bar and Stator Inter-Turn Fault Conditions Using Analytical Modeling," in IEEE Trans. on Magnetics, vol. 54, no. 11, pp. 1-5, Nov. 2018, Art no. 8203705.

[3] Joksimovic, G.; Djurovic, M; Penman, J., "Cage rotor MMF: Winding Function Approach,"Power Engineering Review, IEEE, Vol. 21, No. 4, 64-66.April 2001.

[4] D. Meeker, "Finite element method magnetics: Download," Finite Element Method Magnetics, https://www.femm.info/wiki/Download (accessed Sep. 1, 2023).

[5] D. Meeker, "Finite element method magnetics: Octavefemm - femm.info," Finite Element Method Magnetics, https://www.femm.info/wiki/Files/files.xml?action=download&file=octavefemm.pdf (accessed Sep. 1, 2023).

Kratka biografija:

Milan Vranješ rođen je u Novom Sadu 1997. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine odbranio je 2024.god.

Dejan Jerkan je vanr. prof na Fakultetu tehničkih nauka u Novom Sadu, na Katedri za Energetsku elektroniku i pretvarače. Oblast interesovanja su mu modelovanje i dijagnostika električnih mašina, kao i metoda konačnih elemenata.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.313
DOI: https://doi.org/10.24867/31BE08Vranjes

HARMONICI U SPEKTRU STRUJE TROFAZNE KAVEZNE ASINHRONE MAŠINE

HARMONICS IN THE CURRENT SPECTRUM OF THREE-PHASE SQUIRREL-CAGE INDUCTION MACHINE

Nadica Vranješ, Dejan Jerkan, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Predmet rada je analiza spektra struja kod asinhronih motora sa akcentom na kavezne asinhrone mašine. U ovom radu biće pojašnjeni uzroci nastajanja viših harmonika u kaveznoj asinhronoj mašini sa osvrtom na rad i konstrukciju iste. Analiziraće se spektar statorske struje na primeru trofaznog kaveznog motora nominalne snage* $11kW$*, koji će biti razmatran u režimu nominalnog opterećenja i režimu sa kvarom (oštećenje rotorskih provodnika). U pomenutim režimima ući će se u trag svakoj značajnoj komponenti strujnog spektra tj. navesti njeno poreklo.*

Ključne reči: *Asinhroni motor, Spektar struja, kvarovi*

Abstract *The subject of this paper is the analysis of the current spectrum of asynchronous motors with an emphasis on cage asynchronous machines. In this paper, the stator current spectrum will be analyzed on the example of a squirrel-cage motor with nominal power 11 kW, which will be considered in nominal load mode and failure mode (damage to rotor conductors). In these modes, every significant component of the current spectrum will be traced, i.e. state its origin.*

Keywords: *Induction Motor, Current spectrum, Faults*

1. UVOD

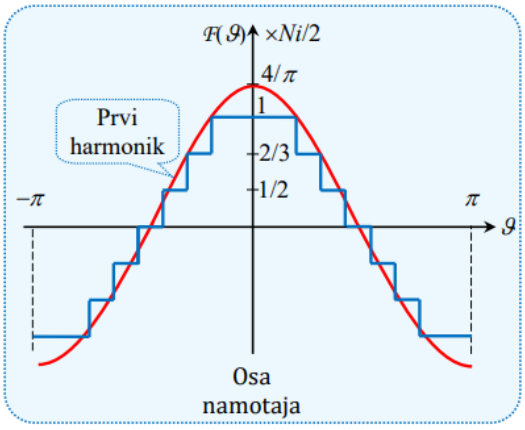
Asinhroni motor kao i svaka električna rotaciona mašina u najgrubljoj podeli sastoji se iz statora i rotora. Stator se izrađuje od feromagnetnog materijala - lameliranjem sa ciljem maksimalnog umanjenja gubitaka u magnetskom kolu. Žlebovi u koje se smeštaju namotaji statora mogu biti otvoreni ili poluzatvoreni, zavisno od snage motora. Kod kaveznog (kratkospojenog) rotora, rotor čine masivni provodnici koji se sa obe strane kratko spajaju takozvanim kratkospojnim prstenovima. Shodno ovakvom načinu izrade nema se električni pristup rotorskom namotaju.
Na slici 1 prikazan je razvijeni oblik moderne izvedbe jedne niskonaponske, trofazne kavezne asinhrone mašine namenjeni za horizontalni tip montaže, koji je ujedno i najčešći u industrijskoj eksploataciji.

Slika 1.1 - *Razvijeni oblik moderne izvedbe niskonaponske trofazne kavezne asinhrone mašine*

Zbog specifičnosti konstrukcije mašine, očekuje se da u harmonijskom spektru struje asinhrone mašine postoje razne komponente, a prilikom nastanka određenih vrsta kvarova javljaju se i dodatne komponente, pa to može da posluži kao sredstvo za monitoring ovih vrsta mašina, što i jeste glavna tema rada.

2. SPEKTAR STATORSKIH STRUJA ISPRAVNE ASINHRONE MAŠINE

U idealnom slučaju kod mašina naizmenične struje cilj je dobijanje prostoperiodične (sinusne) raspodele magnetne indukcije u vazdušnom zazoru. To se kod asinhronih mašina postiže raspodeljivanjem trofaznog statorskog namotaja u žlebove.



Slika 2.1 – *Raspodela magnetnopobudne sile trofaznog statorskog napotaja [4]*

Usled proticanja struje kroz raspodeljeni namotaj uspostavlja se magnetno polje. Ako na cilindričnom magnetnom kolu postoje tri raspodeljena namotaja sa po N

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

navojaka, koji su međusobno prostorno pomereni za $2\pi/3$ i neka kroz njih teku struje koje obrazuju trofazan naizmeničan sistem formiraće se obrtno polje koje je ključno za funkcionisanje asinhronih mašina.
Usled rasporeda trofaznog namotaja u žlebovima statora talasni oblik rotirajuće MPS je stepeničast, stoga ona sadrži pored osnovnog i više prostorne harmonike.
Magnetno kolo mašine je "diskretizovano" konačnim, i u pogledu prostoperiodične prostorne raspodele, vrlo malim brojem žlebova (obično nekoliko desetina) rezultujući, takođe, konačnim brojem položaja u kojem se mogu naći provodnici, što onemogućuje idealnu sinusnu raspodelu. Posledica navedenog činjeničnog stanja odražava se pojavom viših harmonika u prostornoj raspodeli polja. Prisustvo viših harmonika ogleda se u tome da prostorna raspodela magnetne indukcije $B(\theta)$ nije prostoperiodična funkcija ugla (θ).
Ovi prostorni harmonici mogu se opisati poznatim izrazom:

$$v = 6g + 1,\ g = 0, \pm1, \pm2, \dots \quad (2.1)$$

(v) predstavlja red harmonika. Glavni žlebni harmonici (*Pricnipal slot harmonics - PSH*) su dva najistaknutija viša harmonika u stepeničastoj MPS statora.
U slučaju kaveznog asinhronog motora, od mnogo većeg značaja su rotorski žlebni harmonici. Rotorski kavez reaguje na talase magnetne indukcije sa statorske strane sledećim talasima MPS:

$$M_1 = M_{1m}\cos(s_v\omega t - vp\theta_r) \quad (2.2)$$
$$M_2 = M_{2m}\cos\left(s_v\omega t + \left(\frac{\lambda R}{p} - v\right)p\theta_r\right) \quad (2.3)$$
$$M_3 = M_{3m}\cos\left(s_v\omega t - \left(\frac{\lambda R}{p} + v\right)p\theta_r\right) \quad (2.4)$$

gde je Sv:

$$s_v = 1 - v(1 - s) \quad (2.5)$$

U mašini sa nepromenljivom magnetnom otpornošću, pomenuti talasi MPS i konstantna permeabilnost u vazdušnom zazoru u međusobnoj reakciji daju istovetne talase magnetne indukcije, koji gledano sa strane statora izgledaju:

$$B_1 = B_{1m}\cos(\omega t - vp\theta_s) \quad (2.6)$$
$$B_2 = B_{2m}\cos\left(\left(1 - \lambda\frac{R}{p}(1 - s)\right)\omega t + \left(\frac{\lambda R}{p} - v\right)p\theta_s\right) \quad (2.7)$$
$$B_3 = B_{3m}\cos\left(\left(1 + \lambda\frac{R}{p}(1 - s)\right)\omega t - \left(\frac{\lambda R}{p} + v\right)p\theta_s\right) \quad (2.8)$$

Sve prostorne harmonike magnetne indukcije sa statorske strane rotor reflektuje pri baznoj frekvenciji i dva niza frekvencija koje su funkcija klizanja. Ove komponente se nalaze prilično visoko u spektru statorske struje i poznati su kao donji i gornji RSH (*rotor slot harmonic*), respektivno:

$$f_{RSH\_L} = \left(1 - \lambda\frac{R}{p}(1 - s)\right)f_1 \quad (2.9)$$
$$f_{RSH\_U} = \left(1 + \lambda\frac{R}{p}(1 - s)\right)f_1 \quad (2.10)$$

Međutim, postojanje komponenti statorske struje pri navedenim frekvencijama zavisi od broja pari polova p u talasima magnetne indukcije. Preciziije rečeno, egzistencija komponente struje statora uslovljena je time da (v) u $(\frac{R}{p}$ - $v)$ mora da uzima neku vrednost iz skupa

$$H = (6k + 1)\text{ , tj.:} \quad (2.11)$$

$$R_{L\_PSH} = p[6(g + k) + 2], g = 0, \pm1, \pm2, \dots; \quad (2.12)$$
$$k = 0, \pm1, \pm2,\dots$$
$$R_{L\_PSH} = (6n + 2)p,\quad n = 0,1,2,3, \dots \quad (2.13)$$

Slično tome za postojanje gornjeg PSH u spektru statorske struje mora biti zadovoljeno sledeće:

$$R_{U\_PSH} = (6n - 2)p,\quad n = 1,2,3, \dots \quad (2.14)$$

Na kraju za postojanje obe komponente potrebno je da broj šipki rotora u motoru sa p pari polova bude jednak srednjoj vrednosti prethodna dva uslova:

$$R_{BOTH\_PSH} = 6np,\quad n = 1,2,3, \dots \quad (2.15)$$

Rotorski i statorski žlebni efekat je efekat koji se javlja kao posledica promene veličine vazdušnog zazora usled obrtanja rotora - funkcija pozicije rotora.
Rotorski prostorni harmonici će se pojaviti jedino kada struja protiče kroz namotaje kaveza, dok za postojanje rotorskih permeabilnih harmonika ovaj uslov ne mora biti ispunjen, naprotiv ovaj efekat će biti izražen i kada u žlebovima rotora nema namotaja već se njegovo kretanje podstiče nekim spoljašnjim izvorom (npr. pomoćnim motorom). Obrtanje rotora, promena njegove pozicije, u ovom slučaju će takođe imati kao posledicu pojavu komponenti struje definisanih izrazima (2.9) i (2.10) u statorskom namotaju.
U zasićenoj mašini očekuju se nove komponente u struji statora samo na sledećim frekvencijama:

$$f_{S\_L} = \left(3 - \lambda\frac{R}{p}(1 - s)\right)f_1 \quad (2.16)$$
$$f_{S\_U} = \left(3 + \lambda\frac{R}{p}(1 - s)\right)f_1 \quad (2.17)$$

To su niži i viši harmonici zasićenja. Za postojanje ovih harmonika u spektru, neophodno je da bude ispunjen uslov da $R/p - (v + 2)$ pripada $H = (6k + 1)$, $k = 0, \pm1, \pm2$.

3. SPEKTAR NEISPRAVNE ASINHRONE MAŠINE

Kvarovi asinhronih motora se mogu klasifikovati u dve kategorije: mehanički i električni. Mehanički kvarovi obuhvataju oštećenja ležaja i ekscentricitet rotora, dok su tipični električni kvarovi vezani za oštećenja statorskog namotaja – međunavojni kratak spoj i rotora – delimični ili potpuni prekidi štapnih provodnika i kratkospojnih prstenova.
Kvarovi ležaja predstavljaju najčešću vrstu kvarova asinhronih mašina (do 50% svih kvarova). Većina električnih mašina koristi kuglične ili valjkaste ležaje. Oni se sastoje od unutrašnjeg i spoljašnjeg prstena i kuglica/valjčića smeštenih u odgovarajuću kliznu stazu. Oštećenja ležaja kuglične/valjkaste konstrukcije mogu se podeliti na: oštećenja klizne staze spoljašnjeg prstena, oštećenja klizne staze unutrašnjog prstena, oštećenja kuglica ležaja i oštećenja kaveza ležaja. Ova oštećenja stvaraju vibracije specifičnih frekvencija koje su funkcija geometrije ležaja i mogu se izračunati kao:

- Frekvencija vibracije usled oštećenja klizne staze spoljašnjeg prstena:

$$f_{c1} = \frac{N}{2} f_s \left(1 - \frac{D_b}{D_c} cos\theta\right), \quad (3.1)$$

- Frekvencija vibracije usled oštećenja klizne staze unutrašnjeg prstena:

$$f_{c2} = \frac{N}{2} f_s \left(1 + \frac{D_b}{D_c} cos\theta\right), \quad (3.2)$$

- Frekvencija vibracije usled oštećenja kuglica ležaja:

$$f_{c3} = \frac{D_c}{D_b} \frac{1}{2} f_s \left(1 - \left(\frac{D_b}{D_c} cos\theta\right)^2\right), \quad (3.3)$$

- Frekvencija vibracije usled oštećenja kaveza ležaja:

$$f_{c4} = \frac{1}{2} f_s \left(1 - \frac{D_b}{D_c} cos\theta\right). \quad (3.4)$$

Gde je f_{ck} – frekvencija vibracije, f_s – frekvencija rotacije, N – broj kuglica, D_b – prečnik kuglice, D_c – prečnik kaveza ležaja i θ – ugao kontakta kuglice sa stazom. Ove vibracije karakterističnih frekvencija se oslikavaju u pojavi viših harmonika u linijskoj struji motora na frekvencijama:

$$f_c = |f_e \pm m f_c|, m = 1, 2, \ldots \quad (3.5)$$

gde je f_e – frekvencija napajanja.

Međunavojni kratak spoj predstavlja najčešći električni kvar kod asinhronih mašina. Za razliku od kvarova koji obuhvataju navojke dve različite faze ili zemljospojevi koji se mogu lako otkriti jer dovode do naglog povećanja amplitude struje i brzog širenja kroz električno kolo, međunavojni kratki spoj statorskog namotaja u jednoj fazi uglavnom se u začetku neće manifestovati makroskopski i prekostrujna zaštita ga neće blagovremeno otkriti. Ovakva vrsta kvara nije jednostavna za ranu dijagnostiku i najčeće se otkriva tek kada kratak spoj obuhvati značajan deo namotaja. Postoji nekoliko naučno prihvaćenih metoda za neinvazivnu ranu dijagnostiku međunavojnog kratkog spoja u jednoj fazi. Neki od njih obuhvataju spektralnu analizu struja motora, analizu fluksa koristeći spoljašnji i unutrašnji probni namotaj i spektralnu analizu indukovanog napona na priključcima motora nakon njegovog isključenja. Najnovija metoda koja se nameće kao i najpouzdanija do sada obuhvata praćenje putanje vrha polifazora statorskih struja u transformisanom Parkovom području.

Kod asinhronih mašina eksentricitet rotora predstavlja stanje nejednake širine vazdušnog zazora po njegovom obodu. Postoje tri vrste ekscentriciteta vazdušnog zazora: statički, dinamički i mešoviti ekscentricitet. Najčešća tehnika za detekciju eksentriciteta koja se koristi jeste spektralna analiza linijske struje motora. Jedan od najpouzdanijih pokazatelja mešovitog eksentriciteta jeste pojava specifičnih harmonika u okolini mrežnih harmonika čija je frekvencija jednaka:

$$f_{eec} = [m f_e \pm k f_r], \quad (3.6)$$

gde je f_e – osnovna frekvencija napajanja, m – red višeg mrežnog harmonika, f_r – brzina rotacije rotora ($f_r = \frac{1-s}{p} f_e$) i k red harmonika ekscentriciteta koji se pojavljuje u okolini mrežnog harmonika.

Oštećenja štapnih provodnika - oštećenja rotora, mogu dovesti do neželjenih pojava u samom motoru i pogonu, kao što su pulsacije elektromagnetskog momenta i brzine obrtanja motora (što vodi ka ubrzanom habanju ležaja) i stvaranje debalansa rotora koji uzrokuje povišene vibracije motora, a može biti i uzrok stvaranju ekscentriciteta rotora.

Najefikasnija neinvazivna metoda za detekciju kvarova u rotorskom kavezu obuhvata spektralnu analizu struje motora i proveravanje postojanja karakterističnih harmonika – obeležja ove vrste kvara. U slučaju potpunog ili delimičnog prekida štapova u rotoru u struji motora se indukuju subharmonici na specifičnim frekvencijama u okolini svih normalno prisutnih harmonika u spektru struje. Njihove frekvencije se mogu izračunati kao:

$$f_b = [m \pm k2s] f_e, \quad (3.7)$$

gde je f_e – osnovna frekvencija napajanja, m – red višeg harmonika, s – klizanje i k – red harmonika kvara u rotorskom kavezu.

4. ANALIZA SPEKTRA NA REALNOM KAVEZNOM ASINHRONOM MOTORU

Analiza signala struje motora (MCSA) predstavlja jednu od najpopularnijih korišćenih tehnika za detekciju pomenutih kvarova rotora. Prednosti ove metode nad ostalima su: pristupačna i jednostavna merna oprema, merenje se može vršiti iz kontrolnog centra bez direkntog pristupa samom motoru (za razliku od metode merenja vibracija gde se akcelerometar mora fizički ugraditi na motor), nezavisna je u odnosu na parametre motora, merenje struje može se vršiti na bazi Holovog efekta (Holovom sondom) koja za razliku od vrednosti koje daje ampermetar pruža mogućnost snimanja kompletnog talasa struje u vremenskom domenu što je neophodno za dalju analizu (FFT), takođe ovaj način ima širok opseg radnih frekvencija (0-500)Hz, i merenje se vrši u realnom vremenu tokom rada motora tako da sam pogon ne trpi gubitke, a istovremeno ovakav vid merenja otvara vrata prediktivnim i preventivnim održavanjima motora.

Analizira se spektar statorske struje na primeru realnog asinhronog kaveznog motora sa $S = 36, R = 28, 2p = 4$, nominalne snage $11kW$, nominalnog napona $400V$ i $s_n = 3.1\%$ klizanje, koji će biti razmatran u režimu nominalnog opterećenja i režimu sa kvarom (oštećenje rotorskih provodnika). Analiza je urađena u programskom paketu MATLAB.

Prvo se obrađuje slučaj zdravog asinhronog motora pri nominalnom (100%) opterećenju. Komponente u spektru na frekvencijama $25.775Hz$ i $74.225Hz$, , kao i komponente na $728.2Hz$ i $1406.6Hz$, posledica su neuravnoteženosti odnosno ekscentriciteta rotora. One su značajne i pojavile su se u ovom slučaju tako da se ne mogu ignorisati. Prema formuli 1.18 dobijaju se pomenute vrednosti.

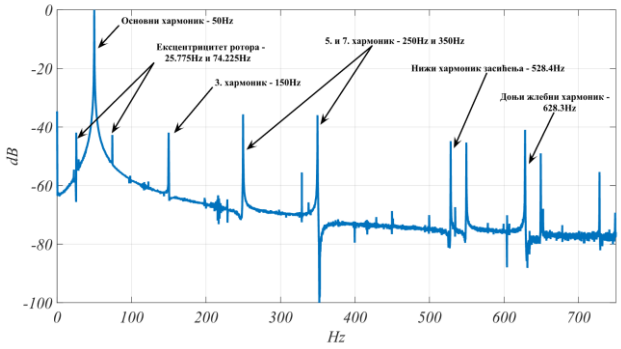
$$f_{ecc} = \left\{(n_{rt} R \pm n_d) \frac{(1-s)}{p} \pm n_v\right\} f_1; \; n_{rt}, n_d, n_v \in \mathbb{Z} \quad (4.1)$$

Izvor napajanja je simetričan, ali usled neuravnoteženosti elementa (asinhronog motora), posledično će se pojaviti sve komponente trećeg umnoška osnovnog harmonika, koje inače (teorijski) ne bi smele da postoje izvan trougla. To su prema formuli 1.19

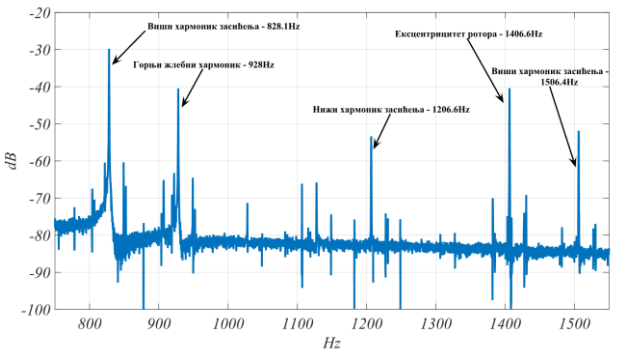
$$f_{usv} = (1 + 2 \cdot k) f_s, k \in \mathbb{Z} \quad (4.2)$$

komponente na: $150Hz, 250Hz, 350\ Hz$ ...

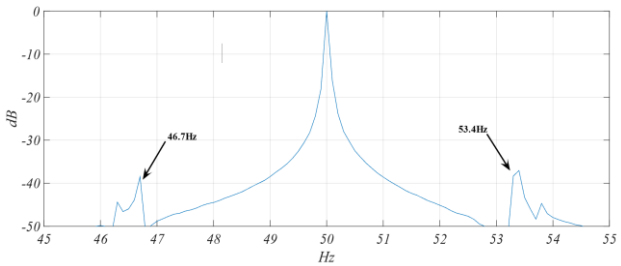
U spektru će se takođe naći i niži $(528.4Hz, 1206.6Hz)$ i viši $(828.1Hz, 1506.4Hz)$ harmonici zasićenja kao i donji žlebni harmonik na $628.3Hz$, ali ne i gornji žlebni zbog broja rotorskih provodnika. U strujnom spektru motora sa oštećenim rotorskim provodnicima (režim sa kvarom), u blizini osnovnog harmonika javljaju se dve komponente $(46.7Hz, 53.4Hz)$ koje ukazuju na navedeni kvar.



Slika 4.1 – *Frekvencijski spektar ispravne mašine od 0 – 750 Hz*



Slika 4.2 – *Frekvencijski spektar ispravne mašine od 750 - 1500 Hz*



Slika 4.3 – *Komponente usled oštećenih rotorskih provodnika*

5. ZAKLJUČAK

U radu je izložena i objašnjena analiza spektra statorskih struja asinhronih kaveznih mašina. Navedeni su uzroci nastanka i posledice viših harmonika kao i načini da se iskoristi njihovo prisustvo u okviru teme rada. Takođe rad se bavio tehnikama akvizicije i obrade signala kao i njegovom prirodom. Na kraju su teoretske osnove primenjene na primeru realnog asinhronog kaveznog motora i priloženi su dobijeni rezultati. Analiza spektra statorskih struja MCSA metodom predstavlja vrlo koristan alat za praćenje rada asinhronih motora i dijagnostikovanje raznih kvarova u ranoj fazi u realnom vremenu kod pomenutih mašina. Navedene prednosti, konkretno neinvazivnost i monitoring u realnom vremenu, čine ovu metodu vrlo primamljivom za implementaciju u velikim pogonima jer omogućava prediktivno održavanje koje ne zahteva prekid rada pogona i time osetno redukuje troškove. Sa razvojem veštačkih neuronskih mreža i veštačke inteligencije otvaraju se vrata i za unapređenje MCSA metode u pravcu kreiranja autonomnog sistema koji bi na osnovu naučenih šablona mogao da prepoznaje anomalije u frekvencijskom spektru i samostalno, bez nadzora čoveka, otkrivao eventualne kvarove na mašini i u skladu sa tim reagovao.

6. LITERATURA

[1] Dr Dejan Reljić: *Otkrivanje kvara rotora kaveznog asinhronog motora primenom tehnika analize terminalnih veličina,* Novi Sad, 2017.

[2] Emil Levi, Vladan Vučković, Vladimir Strezoski: *Osnovi elektroenergetike – elektroenergetski pretvarači,* FTN Izdavaštvo, Novi Sad, 2013.

[3] Gojko Joksimović, Jakša Riger, Thomas Wolbank, Nedjeljko Perić, Mario Vašak: *Stator line current spectrum content of a healthy cage rotor induction machine,* 2011.

[4] Veran Vasić, *Skripta Uvod u električne mašine,* Novi Sad, 2023.

[5] Toliyat, H. A., Nandi, S., Choi, S., & Meshgin-Kelk, H. (2017). *Electric Machines - Modeling, Condition Monitoring, and Fault Diagnosis*

[6] A. Bellini, F. Filippetti, C. Tassoni and G. Capolino, "*Advances in Diagnostic Techniques for Induction Machines,*" in IEEE Transactions on Industrial Electronics, vol. 55, no. 12, pp. 4109-4126, Dec. 2008, doi:10.1109/TIE.2008.2007527.

Kratka biografija:

Nadica Vranješ rođena je u Novom Sadu 1997. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine odbranila je 2024.god.

Dejan Jerkan je vanr. prof. na Fakultetu tehničkih nauka u Novom Sadu, na Katedri za Energetsku elektroniku i pretvarače. Oblast interesovanja su mu modelovanje i dijagnostika električnih mašina, kao i metoda konačnih elemenata.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.4
DOI: https://doi.org/10.24867/31BE11Vukovic

INTERAKTIVNI GRAFIČKI EDITOR ZA EVALUACIJU GARBLED CIRCUITS PROTOKOLA

INTERACTIVE GRAPHICAL EDITOR FOR GARBLED CIRCUITS PROTOCOLS EVALUATION

Zorica Vuković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – SOFTVERSKO INŽENJERSTVO

Kratak sadržaj – *U ovom radu je predstavljena oblast bezbednog izračunavanja sa više učesnika, kao i Yao's Garbled Circuits protokol koji predstavlja njenu osnovnu implementaciju. Opisani su sigurani načini razmene privatnih podataka pomoću Oblivious Transfer protokola. Predstavljeni protokoli su evaluirani kroz interaktivni grafički editor za evaluaciju logičkih kola koji usvaja prethodne koncepte.*

Ključne reči: *MPC, 2PC, Yao's GC protokol, logička kola, interaktivni grafički editor*

Abstract – *This paper presents the field of secure multi-party computation and Yao's Garbled Circuits protocol, which represents its basic implementation. It also descibes a secure ways of exchanging private data using the Oblivious Transfer protocols. The presented protocols were evaluated through an interactive graphical editor for the evaluation of logic circuits that adopts the previous concepts.*

Keywords: *MPC, 2PC, Yao's GC protocol, OT protocol, boolean circuit, interactive graphic editor*

1. UVOD

U vremenu konstantnog razvoja tehnologije, razmena podataka putem mreže postaje sve češća. Ipak, privatnost podataka koji se koriste prilikom različitih proračuna se sve više dovodi u pitanje. Manipulacije rezultatima aukcija i otkrivanje poverljivih ponuda, samo su neki od problema do kojih može doći ukoliko izračunavanja nad privatnim podacima nisu sigurna. U takvom okruženju, gde međusobno poverenje učesnika nije zagarantovano, javlja se potreba za razvojem mehanizama koji omogućavaju zajedničku obradu podataka uz zaštitu osetljivih informacija.

Zamislimo da dve osobe žele da saznaju ko ima više novca, ali bez otkrivanja svojih iznosa. Ovaj problem, poznat kao problem dva milionera (engl. *Two Millionaires Problem*) [1], zahteva računanje sa privatnim podacima uz očuvanje poverljivosti.

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

Tradicionalni načini rešavanja ovog problema su podrazumevali angažovanje treće osobe od poverenja ili upotrebu različitih neefikasnih tehnika šifrovanja, što je zahtevalo velike resurse. Zbog toga je razvoj oblasti koja omogućava bezbedno i efikasno računanje nad osetljivim podacima od ključne važnosti za napredovanje mrežne bezbednosti.

Jedan od mehanizama koji teži da reši navedene probleme predstavlja grupa protokola za bezbedno izračunavanje sa više učesnika (engl. MPC - *Multi-Party Computation*) [2]. Primenjujući MPC protokole, obezbeđuje se privatnost podataka svih učesnika u različitim računskim procesima. U ovom radu posebna pažnja posvećena je Jaovom *Garbled Circuits* protokolu (Yao's GC), koji je od suštinskog značaja za razvoj MPC oblasti [3]. Ovaj protokol omogućava evaluaciju funkcija izraženih logičkim kolima u kojima interaguju dva učesnika (2PC). U radu će biti opisani i tipovi *Oblivious Transfer* (OT) potprotokola, koji igraju vitalnu ulogu u obezbeđivanju sigurnog prenosa podataka između učesnika u procesu evaluacija funkcija [4]. Takođe, kao glavni doprinos rada, evaluacija upotrebe opisanih protokola će biti urađena u interaktivnom grafičkom editoru koji igra ključnu ulogu u vizualizaciji i istraživanju sigurnosnih aspekata računanja, pružajući korisnicima intuitivno i efikasno okruženje za analizu i simulaciju kompleksnih logičkih kola.

2. BEZBEDNO IZRAČUNAVANJE SA VIŠE UČESNIKA

2.1. Definicija MPC

MPC je oblast kriptografije koja omogućava sigurno računanje u distribuiranim sistemima, efikasno štiteći podatke od zlonamernih učesnika. Osnovna definicija bezbednog izračunavanja se formalizuje kroz idealni svet. U idealnom svetu, učesnici šalju privatne podatke pouzdanoj trećoj strani, koja računa željenu funkciju i vraća rezultate. Iako ovaj pristup garantuje bezbednost, postojanje pouzdane treće strane ga čini imaginarnim. Umesto toga, učesnici međusobno komuniciraju koristeći unapred definisan protokol koji obezbeđuje da rezultati budu isti kao u idealnom svetu.

Prilikom razmatranja MPC važno je i precizno definisati ponašanje učesnika prilikom izračunavanja funkcija. Upravo tome služe dva osnovna modela bezbednosti: pasivni (engl. *semi-honest*) i aktivni (engl. *malicious*) [5].

U pasivnom modelu učesnici prate protokol, ali pokušavaju da naruše privatnost drugih učesnika. Sa druge strane, u aktivnom modelu bezbednosti, učesnici mogu da odstupe od protokola ili šalju pogrešne podatke kako bi narušili izračunavanje funkcije.

2.2. Osnovni MPC protokoli

Na temelju MPC oblasti su izgrađeni mnogi protokoli, a samo neki od njih su prikazani u Tabeli 1. Kako je prilikom odabira protokola važno da protokol bude što efikasniji, podaci o broju rundi i količini podataka koju je potrebno preneti prilikom evaluacije su prvi koji se upoređuju. Protokoli sa manjim brojem rundi i minimalnim komunikacionim troškovima su poželjniji, posebno u kompleksnim sistemima. Na primer, *Beaver-Micali-Rogaway* (BMR) protokol [11], koji koristi konstantan broj rundi, predstavlja značajan napredak u ovoj oblasti.

Tabela 1. *Osnovni MPC protokoli [5]*

Protokol	Broj učesnika	Broj rundi	Tip kola
Yao's GC	2	konstantan	logičko
GMW [12]	više od 2	dubina kola	logičko / aritmetičko
BGW [13]	više od 2	dubina kola	logičko / aritmetičko
BMR	više od 2	konstantan	logičko
GESS [14]	2	konstantan	logička formula

Oblast koja se razlikuje od opšteg slučaja sa više učesnika je MPC sa dva učesnika (engl. 2PC - *Two-Party Computation*) [6]. U ovom scenariju, dva učesnika mogu zajedno da izračunaju funkciju bez otkrivanja svojih privatnih podataka. Razvijen je veliki broj protokola koji omogućavaju ovaj oblik sigurne komunikacije, a jedan od najpoznatijih je Yao's GC protokol.

3. YAO'S GC PROTOKOL

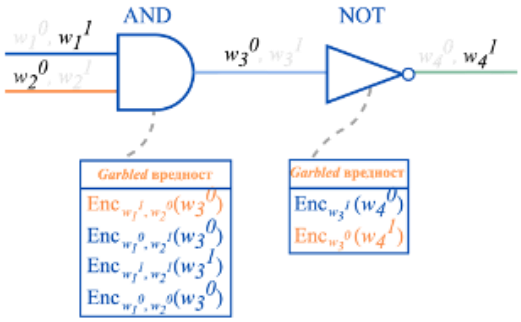
3.1. Klasičan Yao's GC protokol

Yao's GC protokol je jedan od najaktivnije proučavanih MPC metoda, inicijalno razvijen da podrži dva učesnika: *garbler*, u nastavku Alisa, koja formira *garbled* kolo i *evaluator*, odnosno Bob, koji izračunava izlaznu vrednost logičkog kola.

Osnovni koraci klasičnog Yao's GC protokola su sledeći:

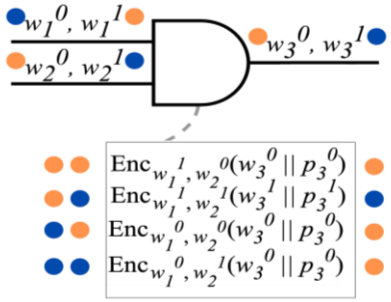
1) Alisa generiše logičko kolo za željenu funkciju f.
2) Alisa formira *garbled* kolo na osnovu logičkog kola tako što se za svaku logičku kapiju (engl. *gate*) šifruje tablica istinitosti. Ovaj proces obuhvata šifrovanje svakog podatka iz tablice istinitosti korišćenjem slučajno generisanih ključeva. Na samom kraju, redovi u formiranoj tabeli se permutuju kako se izlazna vrednost ne bi mogla odrediti na osnovu reda u tabeli.
3) Alisa šalje Bobu formirano *garbled* kolo, zajedno sa šifrovanim vrednostima svojih ulaza.
4) Bob pomoću OT protokola od Alise dobija podatke o šifrovanim vrednostima svojih ulaza, bez otkrivanja privatnih podataka. Opis OT protokola je prikazan u narednoj sekciji 3.2.
5) Bob evalira logičko kolo tako što prolazi kroz sve logičke kapije i pokušava da dešifruje redove u njihovim garbled tabelama. Ukoliko je protokol uspešno sproveden, Bob za svaku tabelu uspeva da pročita tačno jedan red.

Uprošćen prikaz evaluacije logičkog kola je dat na slici 1.



Slika 1. *Prikaz evaluacije logičkog kola. Ulazna vrednost koju unosi Alisa na žici w_1 je 1, dok je Bob na žicu w_2 uneo 0. Prvi red u levoj i drugi red u desnoj garbled tabeli su označeni kao uspešno dešifrovani redovi.*

Kako prethodno opisan klasičan Yao's GC protokol nije efikasan u realnim uslovima, vremenom su se razvile različite tehnike optimizacije [7, 8]. One se fokusiraju na smanjenje veličine *garbled* kola i broja funkcija koje se pozivaju tokom evaluacije. Jedna od ključnih tehnika optimizacije je *point-and-permute* [8]. Ona omogućava Bobu da utvrdi koju šifrovanu vrednost treba da dešifruje, bez potrebe da dešifruje sve četiri vrednosti kao kod klasičnog protokola. Ovo se postiže dodavanjem nasumičnog selekcionog bita svakoj žici. Na slici 2 se vidi da šifrovane vrednosti za ulaze imaju pridružene slučajne bitove. Oni služe da se na osnovu njih indeksira tablica istinitosti. Na ovaj način se smanjuje broj nepotrebnih operacija i ubrzava proces jer Bob tokom evaluacije tačno zna koji red treba da dešifruje.



Slika 2. *Prikaz formiranja garbled kola korišćenjem point-and-permute tehnike sa p selekcionim bitom*

3.2. OT protokol

U mnogim protokolima bezbednog izračunavanja postoji potreba da jedna strana dobije informacije od druge bez otkrivanja suvišnih podataka. Ovo proizilazi iz činjenice da je danas malo prostora za potpuno poverenje između učesnika prilikom komunikacije. Zbog toga je potreban protokol koji će omogućiti diskretnu razmenu informacija. OT pruža siguran način razmene podataka uz minimalno otkrivanje informacija [9].

U osnovi OT protokol ima tri varijante [4]:

- *1-out-of-2* je verzija u kojoj učestvuju dve strane. Primalac na kraju protokola saznaje jednu od dve

ulazne vrednosti koje poseduje pošiljalac, pri čemu pošiljalac ne zna ništa o izboru primaoca.

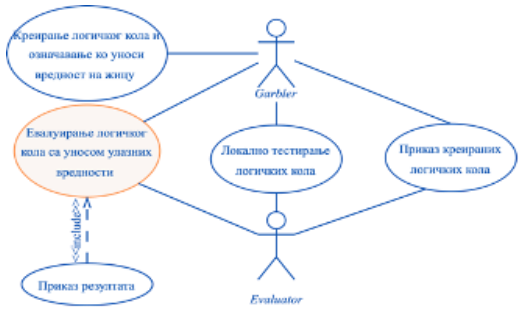
- *1-out-of-n* je proširena verzija *1-out-of-2* OT protokola koja omogućava jednoj strani, odnosno primaocu da dobije jedan od nekoliko podataka koje poseduje druga strana. Takođe, pošiljalac ne dobija informaciju o izboru primaoca.
- *k-out-of-n* omogućava jednoj strani, odnosno primaocu da dobije više podataka od druge strane, pri čemu pošiljalac ima nekoliko ulaznih podataka. Na kraju protokola, primalac saznaje onoliko vrednosti koliko je izabrao, dok pošiljalac ne dobija nikakvu informaciju o izboru primaoca.

4. SPECIFIKACIJA GRAFIČKOG EDITORA

Cilj interaktivnog grafičkog editora za evaluaciju *Garbled Circuits* protokola je da korisnicima omogući intuitivno kreiranje i analizu logičkih šema. Ovako kreirane logičke šeme se koriste u protokolu bezbednog izračunavanja sa dva učesnika.

4.1. Specifikacija funkcionalnosti

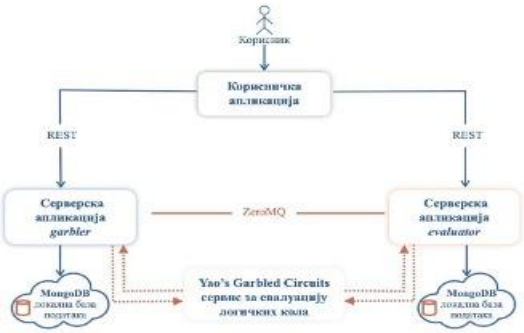
Dijagram slučajeva korišćenja je najjednostavniji način specifikacije šta određeni učesnici mogu da izvrše u posmatranom sistemu. Na slici 3 je prikazan dijagram slučajeva korišćenja za editor evaluacije logičkih kola, koji obuhvata dve ključne uloge: *garbler* i *evaluator*. *Garbler* može da kreira logičko kolo i bira učesnika koji će tokom evaluacije da unosi vrednost na datu žicu. Obe uloge mogu da pregledaju i lokalno testiraju logičko kolo, a nakon toga *evaluator* može da započne evaluaciju logičkog kola. Takođe, učesnici imaju i mogućnost pregledanja dobijenih rezultata evaluacije.



Slika 3. *Dijagram slučajeva korišćenja*

4.2. Specifikacija sistema

Glavne komponente interaktivnog grafičkog editora za evaluaciju *Garbled Circuits* protokola su prikazane na slici 4.



Slika 4. *Arhitektura interaktivnog grafičkog editora za evaluiraciju Garbled Circuits protokola*

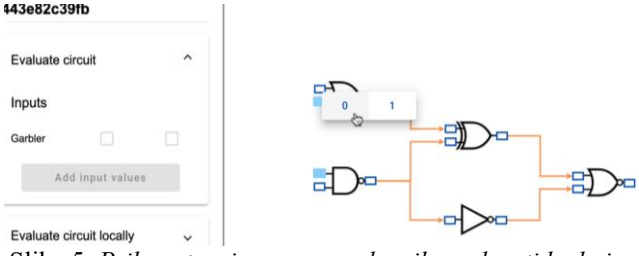
Korisničku interakciju sa sistemom omogućava aplikacija implementirana u *Angular 18* radnom okviru. Fokus ove aplikacije je na formiranju okruženja koje će korisnicima ponuditi jednostavno kreiranje, pregled i evaluaciju logičkih kola. Nakon odabira željene funkcionalnosti, poziva se metoda određenog servisa serverske aplikacije, a komunikacija se odvija putem HTTP protokola.

Serverske aplikacije su implementirane pomoću *Spring Boot 3* radnog okvira. *Garbler* serverska aplikacija predstavlja modul za kreiranje *garbled* verzije logičkog kola, koja se čuva u lokalnoj *MongoDB* bazi podataka. Server za evaluiranje kola komunicira sa Yao's GC servisom, omogućavajući izračunavanje logičkog kola bez otkrivanja osetljivih informacija. Komunikacija između servera se u realnom vremenu, bez dodatnog nadzora mreže, odvija preko *ZeroMQ* sistema poruka [10].

Nakon što je opisan proces razmene podataka između komponenti, važno je naglasiti da sistem treba da omogući jednostavno dodavanje novih protokola za izračunavanje logičkih kola, bez remećenja postojećih funkcionalnosti. Iz tog razloga, arhitektura sistema je dizajnirana da bude proširiva, sa protokolima implementiranim u zasebnim modulima koji komuniciraju putem zajedničkog interfejsa. Ovakav pristup obezbeđuje fleksibilnost sistema i olakšava buduće nadogradnje bez ugrožavanja stabilnosti.

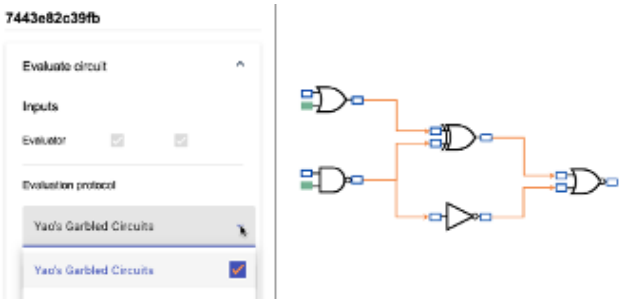
5. IMPLEMENTACIJA GRAFIČKOG EDITORA

Osnovna funkcionalnost interaktivnog grafičkog editora za evaluaciju *Garbled Circuits protokola* je evaluacija izabranog logičkog kola. Proces počinje unosom binarnih vrednosti na ulazne žice prvog sloja kapija, pritiskom na pravougaonike i izborom vrednosti. Stranica je prikazana na slici 5.



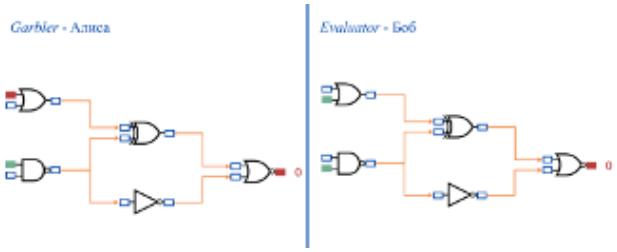
Slika 5. *Prikaz stranice za unos ulaznih vrednosti kada je pregleda garbler*

Kada Alisa uspešno unese vrednosti na ulazne žice, omogućava se pokretanje evaluacije logičkog kola. Tada Bob dobija mogućnost da učestvuje u evaluaciji logičkog kola. Na slici 6 je prikazana stranica za unos i pokretanje evaluacije logičkog kola kada je pregleda *evaluator*. Proces dodavanja ulaznih vrednosti se obavlja na isti način kao što to radi i Alisa. Bob, za razliku od Alise, ima opciju da bira po kom protokolu će biti vršeno evaluiranje datog logičkog kola. Trenutno, sistem podržava evaluaciju korišćenjem klasičnog Yao's GC protokola koji je opisan u poglavlju 3.1.



Slika 6. *Prikaz stranice za slanje unetih ulaznih vrednosti kada je pregleda evaluator*

Po završetku evaluacije logičkog kola, oba učesnika dobijaju obaveštenje o tome, pri čemu nijedna strana na kraju nije svesna ulaznih vrednosti koje je suprotna strana odabrala. Poređenje stranica za prikaz rezultata kod oba učesnika se nalazi na slici 7.



Slika 7. *Prikaz rezultata uspešnog evaluiranja logičkog kola*

6. ZAKLJUČAK

U ovom radu je predstavljeno rešenje za kreiranje interaktivnog grafičkog editora za evaluaciju *Garbled Circuits* protokola. Kako je akcenat na sigurnosti podataka sve značajniji, odatle proizilazi potreba da se omogući grupi nezavisnih učesnika koji ne veruju jedni drugima ili bilo kojoj trećoj strani, da bezbedno izračunaju funkciju koja zavisi od njihovih privatnih ulaza. Ovaj editor predstavlja rešenje koje olakšava vizualizaciju i manipulaciju logičkim kolima, istovremeno implementirajući principe bezbednog izračunavanja.

U radu su prvo opisane teorijske osnove MPC oblasti i predstavljen Yao's GC protokol. Opisani su i OT protokoli, koji omogućavaju primaocu da dobije jednu od ulaznih vrednosti koje poseduje druga strana, dok pošiljalac ne saznaje nikakve informacije o izboru primaoca. Na ovaj način je ukazano na mogućnost stvaranja sigurnog digitalnog ekosistema za obradu logičkih kola, gde su svi podaci zaštićeni tokom prenosa i obrade. Ovakav sistem obezbeđuje saradnju između učesnika bez rizika od narušavanja privatnosti. Specifikacija sistema je prikazana u nastavku rada, dok je na kraju opisana konkretna implementacija funkcionalnosti evaluiranja logičkog kola.

Koraci daljeg razvoja platforme za evaluaciju kola uključuju implementaciju različitih tehnika optimizacije klasičnog Yao's GC протокола, kao i drugih protokola koji koristi logička kola. Takođe, ukoliko bi se interaktivni grafički editor posmatrao kao mesto daljeg razvoja, može se razmotriti kreiranje editora koji bi obezbedio više mogućnosti za manipulaciju logičkim kapijama i žicama između njih. Ovo bi korisnicima omogućilo veću fleksibilnost u kreiranju i modifikaciji logičkih šema, čime bi se unapredila njihova interakcija sa platformom.

7. LITERATURA

[1] Boudot, F., Schoenmakers, B., & Traore, J. (2001). A fair and efficient solution to the socialist millionaires' problem. Discrete Applied Mathematics, 111(1-2), 23-36.
[2] Lindell, Y. (2020). Secure multiparty computation. Communications of the ACM, 64(1), 86-96.
[3] Lindell, Y., & Pinkas, B. (2009). A proof of security of Yao's protocol for two-party computation. Journal of cryptology, 22, 161-188.
[4] Yadav, V. K., Andola, N., Verma, S., & Venkatesan, S. (2022). A survey of oblivious transfer protocol. ACM Computing Surveys (CSUR), 54(10s), 1-37.
[5] Evans, D., Kolesnikov, V., & Rosulek, M. (2018). A pragmatic introduction to secure multi-party computation. *Foundations and Trends® in Privacy and Security*, *2*(2-3), 70-246.
[6] Malkhi, D., Nisan, N., Pinkas, B., & Sella, Y. (2004, August). Fairplay-Secure Two-Party Computation System. In USENIX security symposium (Vol. 4, p. 9).
[7] Yakoubov, S. (2017). A gentle introduction to yao's garbled circuits. Dostupno na https://web. mit. edu/sonka89/www/papers/2017ygc.pdf (datum pristupa 10-09-2024)
[8] Beaver, D., Micali, S., & Rogaway, P. (1990, April). The round complexity of secure protocols. In Proceedings of the twenty-second annual ACM symposium on Theory of computing (pp. 503-513).
[9] Rabin, M. O. (2005). How to exchange secrets with oblivious transfer. Cryptology ePrint Archive.
[10] Hintjens, P. (2013). *ZeroMQ: Messaging for Many Applications*. O'Reilly Media.
[11] Beaver, D., Micali, S., & Rogaway, P. (1990, April). The round complexity of secure protocols. In Proceedings of the twenty-second annual ACM symposium on Theory of computing (pp. 503-513).
[12] Goldreich, O., Micali, S., & Wigderson, A. (2019). How to play any mental game, or a completeness theorem for protocols with honest majority. In Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali (pp. 307-328).
[13] Ben-Or, M., Goldwasser, S., & Wigderson, A. (2019). Completeness theorems for non-cryptographic fault-tolerant distributed computation. In Providing sound foundations for cryptography: on the work of Shafi Goldwasser and Silvio Micali (pp. 351-371).
[14] Kolesnikov V., (2005). Gate evaluation secret sharing and secure one-round two-party computation, Advances in Cryptology-ASIACRYPT.

Kratka biografija:

Zorica Vuković rođena je u Sremskoj Mitrovici 2000. godine. Osnovne akademske studije je završila 2023. godine na Fakultetu tehničkih nauka u Novom Sadu. Master rad na Fakultetu tehničkih nauka iz oblasti Softversko inženjerstvo – Elektronsko poslovanje odbranila je 2024. godine.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.9
DOI: https://doi.org/10.24867/31BE12Karanovic

MAGIČNA AKCIONO-AVANTURISTIČKA VIDEO IGRA ZA MOBILNE UREĐAJE

MAGICAL ACTION-ADVENTURE VIDEO GAME FOR MOBILE DEVICES

Bojana Karanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Ovaj rad opisuje razvoj igre Potion Master: Free the Ghosts, magične akciono-avanturističke igre za mobilne uređaje. Rad detaljno opisuje priču, likove, gejmplej, mehanizme igre, dizajn nivoa, sisteme igre i korisnički interfejs.*

Ključne reči: *Razvoj igre, magična akciona avantura, mobilne igre, Potion Master: Free the Ghosts*

Abstract – *This paper describes the development of the game Potion Master: Free the Ghosts, a magical action-adventure game for mobile devices. The paper provides a detailed description of the story, characters, gameplay, game mechanics, level design, game systems, and user interface.*

Keywords: *Game development, magical action adventure, mobile games, Potion Master: Free the Ghosts.*

1. UVOD

Mobilne video igre su se razvile kao jedan od najpopularnijih oblika zabave u svetu, privlačeći igrače svih uzrasta [1]. Sa svakodnevnim napredovanjem tehnologije i poboljšanjem korisničkog iskustva, mobilne igre su postale ne samo pristupačne, već i vizuelno impresivne i intuitivne. Nakon što su se utvrdile u industriji, ove igre ponudile su nove načine za razonodu, socijalizaciju i istraživanje kreativnosti, stvarajući dinamično okruženje u kojem igrači mogu da uživaju u interaktivnom sadržaju u svakom trenutku i na svakom mestu.

Potion Master: Free the Ghosts je 3D akciono-avanturistička igra prilagođena mobilnim uređajima, namenjena deci avanturistima i ljubiteljima magije. Igrači preuzimaju ulogu mlade čarobnice koja živi u maloj kolibi u šumi, specijalizovanoj za pravljenje magičnih napitaka. Kada duh iz daleke šume moli za njenu pomoć, čarobnica se upušta u opasnu misiju spasavanja zarobljenih duhova od zlih paukova.

Igra *Potion Master: Free the Ghosts* usmerena je ka deci u dobi od 8 do 12 godina. *Potion Master: Free the Ghosts* je dizajnirana da privuče mlade igrače kroz svoju magičnu avanturu i interaktivni svet. Vedre, jasne boje i jednostavan stil likova i okruženja čine igru vizuelno atraktivom i lako razumljivom. Zvučni efekti i prijatna,pomalo čarobna muzika doprinose magičnom raspoloženju igre, što dodatno privlači ovu demografsku grupu. Interaktivnost okruženja omogućava igračima da istražuju šumu, prikupljaju magične sastojke i stvaraju napitke, pružajući im dublji uvid u svet čarolija. Osim pružanja zabave, igra *Potion Master: Free the Ghosts* promoviše pozitivne vrednosti kao što su hrabrost, upornost i prijateljstvo, što je inspirativno i obrazovno za mlade igrače.

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

2. PRIČA IGRE

Priča igre prati mladu čarobnicu, koja živi u kolibi blizu začarane šume, koja je posvećena pravljenju magičnih napitaka i pomaganju stanovnicima okolnih sela. Iako je talentovana, često sumnja u svoje sposobnosti. Jednog dana, dok je sakupljala magične pečurke, susreće duha po imenu Grinbel, koji je pobegao iz daleke šume napadnute od strane zlih paukova. Grinbel moli čarobnicu da mu pomogne da oslobodi zarobljene duhove i povrati ravnotežu u svetu energije. Čarobnica isprva odbija, bojeći se svojih ograničenih moći, ali nakon što joj Grinbel obeća podršku i pomoć u jačanju njenih veština, ona odlučuje da prihvati izazov.

Zajedno kreću na opasno putovanje kroz šumu, suočavajući se sa paukovima, prikupljajući magične sastojke i praveći napitke potrebne za oslobađanje duhova. Najveći izazov čeka ih u centru šume, gde je zarobljen starešina duhova, kojeg čuvaju najmoćniji pauci. Nakon teške borbe i trenutaka očaja, čarobnica otkriva skrivenu snagu u sebi i uspeva da pobedi paukove i oslobodi starešinu.

Starešina duhova joj daje knjigu sa mudrošću svih duhova koje je oslobodila, zahvaljujući joj na pomoći. Dok se vraća kući, čarobnica prolazi kroz lavirint mračne šume, gde pomoću magičnog zapisa iz knjige priziva mačku koja joj pomaže da pronađe izlaz. Vraćajući se u svoju kolibu, ona oseća novo samopouzdanje i hrabrost, spremna za buduće avanture.

3. LIKOVI

U ovom poglavlju će biti detaljno opisani likovi u igri, počevši od protagoniste igre, nakon koga će biti opisan mentor i antagonisti.

3.1. Protagonista

Čarobnica je protagonista igre, slika 1. Živi u blizini magične šume. Pomaže svim bićima u okolini. Nedostaje joj iskustvo ali je spremna da nauči. Veoma je hrabra. Ona je mlada devojka nasmejanog lica i rumenih obraza. Kao i svaka čarobnica, nosi šešir, ljubičaste boje. Haljina

joj je takođe ljubičasta, a kosa duga i crna. Čarobnica može da pravi napitke i da ih aktivira.

Slika 1. *Izgled modela protagoniste*

3.2. Mentor

Grinbel, slika 2, je mentor protagonistkinji igre. Grinbel je mladi duh iz čarobne šume. Uspeo je da pobegne tokom napada zlih paukova i sada traži pomoć kako bi oslobodio ostale duhove. Pošto može da oseti zarobljene duhove, pomaže čarobnici da ih pronađe i oslobodi.

Grinbel je beli, transparentni duh. Ima zelenu kapicu, te otuda i ime. Ima male ruke i velike crne oči.

Slika 2. *Izgled modela mentora*

3.1. Antagonista

Pauci su antagoniste igre, slika 3. Pauci su zli. Napali su i zarobili duhove kako bi oni mogli da vladaju energetskim svetom. Pauci žele da vladaju energetskim svetom kako bi iskoristili njegovu moć za svoje mračne ciljeve. Oni veruju da će steći apsolutnu kontrolu nad magičnim energijama zarobivši duhove koji održavaju energiju u tom svetu. Pauci su veliki i opasni. Mogu biti crni ili smeđi. Imaju sposobnost da pronađu čarobnicu i da napadaju u vidu gađanja otrovom, koji oslabljuje protivnike.

Slika 3. *Izgled modela antagonsite*

4. RAZVOJ MODELA

Kako je tokom igranja igre protagonist u prvom planu i uvek vidljiv, ima smisla objasniti ideje i sam razvoj modela. Model je razvijan pomoću softvera za 3D modelovanje, Blender. Ovaj proces je uključivao više faza, od inicijalne ideje i koncepta do tehničke realizacije i pripreme za animaciju.

4.1. Ideja i reference

Imajući u vidu da je igra namenjena mlađoj publici, dizajn protagonistkinje je bio vođen tom idejom. Protagonistkinja je zamišljena kao dobra čarobnica sa prijateljskim izgledom, koja treba da bude pristupačna i prijatna mladim igračima. Kao osnova za dizajn korišćene su reference i vizuelne smernice inspirisane animacijama i igrama za decu. Karakter ima čovekoliki oblik, sa naglašenim proporcijama – velika glava i krupne oči u odnosu na ostatak tela, što stvara simpatije kod mlađih igrača.

4.2. Modelovanje karaktera

Modelovanje karaktera je započeto modelovanjem torzoa, odnosno haljine. Za torzo je korišćena kocka, dok su ruke, noge i vrat modelovani pomoću cilindra. Šake su modelovane takođe pomoću kocke, a zbog jednostavnosti modela postoje samo četiri prsta. Modelovanje glave je zahtevalo najviše pažnje, dok je pravljenje udubljenja za oči i usta, kao i ispupčenja za nos predstavljalo izazov. Oči su modelovane kao tri kocke, koje su modifikatorima učinjene da izgledaju kao sfere, kako bi se osigurala pravilna i čista geometrija.

5. GEJMPLEJ

U igri *Potion Master: Free the Ghosts,* igrač istražuje 3D prostor, krećući se kroz veliki svet igre. Igrač može da trči, skače i interaguje sa okruženjem. Kada je igrač neaktivan, čarobnica maše, dozivajući ga da nastavi igru.

Pored osnovnog kretanja, igrač može da sakuplja sastojke i pravi različite napitke koji će mu pomoći u oslobađanju duhova. Igrač može sakupljati sastojke i kombinovati ih u kazanu kako bi stvorio napitke. Na primer, napitak za ubrzanje omogućava čarobnici da se brže kreće određeni vremenski period, dok napitak za lečenje trenutno obnavlja njeno zdravlje ako je povređena.

Osim ovih, postoji i napitak za napad koji čarobnici omogućava da uništi neprijatelje koji joj stoje na putu. Da bi napravio napitke, igrač mora da prikupi dovoljno sastojaka i pažljivo ih kombinuje. Napici se mogu koristiti u različitim trenucima igre, pomažući igraču da lakše reši izazove.

Cilj igre je da se spasu svi zarobljeni duhovi. Da bi to postigao, igrač mora da pronađe sastojke razbacane po celoj mapi, napravi odgovarajući napitak i donese ga duhu. Pored sakupljanja sastojaka i pravljenja napitaka, igrač može da naiđe na neprijatelje, čiji je cilj da ga uništi. Ako neprijatelj nanese dovoljno štete i svede zdravlje čarobnice na nulu, igrač gubi.

U izazove spadaju pronalaženje zarobljenog duha, sam proces sakupljanja sastojaka i pravljenje napitaka, kao i izbegavanje i/ili borba sa neprijateljem. Nagrade se dobijaju nakon pobede nad neprijateljem, kao i nakon oslobađanja duha. Kako nivoi napreduju, nagrade su sve vrednije, ali i sve ređe.

Pored nagrada, prisutne su i kazne. Igrač gubi zdravlje ukoliko ga pogodi otrov neprijatelja. Ukoliko izgubi svo zdravlje, svoju avanturu ponovo počinje od kolibe. Kako je mapa prilično velika, ovo može biti ponekad oštra kazna za igrača, ali i velika motivacija da postane što bolji.

6. MEHNIZMI

Mehanizam igre *Potion Master: Free the Ghosts* sastoji se od osnovnih (core) i alternativnih (satellite) mehanizama, koji zajedno formiraju interaktivnu i dinamičnu igru u kojoj igrač upravlja čarobnicom kako bi rešavao zagonetke, borio se sa neprijateljima i oslobađao duhove.

6.1. Core mehanizmi

Osnovni mehanizmi u igri predstavljaju lokomotivni sistem sa dodatkom bacanja napitaka. Lokomotivni sistem omogućava čarobnici da interaguje sa okruženjem na raznovrsne načine, obuhvatajući osnovne pokrete kao što su stajanje, hodanje, trčanje i skakanje. Pored ovih osnovnih pokreta, igra uključuje i bacanje napitaka koje čarobnici omogućava da se bori protiv neprijatelja.

6.2. Satellite mehanizmi

Pored osnovnih mehanizama postoje i alternativni mehanizmi. Mehanizmi unapređenja omogućavaju igraču da poboljša performanse. Napitak za ubrzanje omogućava čarobnici da dobije na brzini i brže pređe delove mape.

7. DIZAJN NIVOA

Nivoi nisu striktno definisani u igri, ali bi se moglo reći da nivo predstavlja proces oslobađanja duha. Oslobađanjem duha bi se završavao jedan, a započinjao novi nivo igre, gde bi bilo potrebno osloboditi drugog duha.

Ne postoji definisana putanja kuda igrači mogu ili moraju da se kreću, već im je prepuštena potpuna sloboda istraživanja otvorenog sveta. Svet je dizajniran da ima različite oblasti, poput mračne šume, crvene šume, polja cveća, itd.

Tokom svakog nivoa su prisutni neprijatelji. Kako nivoi napreduju, tako je neprijatelja sve više. Takođe, sastojci koje igrač može da pokupi su prisutni tokom svih nivoa.

7.1. Dizajn prvog nivoa

Dizajn prvog nivoa je izuzetno bitan zato što predstavlja prvi kontakt igrača sa igrom. Njegova uloga je da uvede igrača u mehanizam igre, mogućnosti i izazove. Nivo je često dizajniran da bude dovoljno jednostavan kako bi se izbegla frustracija kod igrača na samom početku. Kako je ovo igrica namenjena mlađem uzrastu, ovo je uzeto u razmatranje tokom dizajniranja.

Prvi nivo je dizajniran da bude lak i zanimljiv. Igrač se upoznaje sa osnovnim mehanizmima, koji su generalno dobro poznati. Čarobnica se nalazi kod svoje kolibe, odakle i započinje svaki nivo. Pored kolibe se nalazi kazan u kojem je moguće spremati napitke. U neposrednoj blizini kolibe se nalaze sastojci koje čarobnica može da prikupi. Ovo motiviše igrača da se pokrene i istražuje.

Takođe, tokom prvog nivoa postoji i jedan neprijatelj. Neprijatelj upoznaje igrača sa sistemom kažnjavanja. Oduzimanjem zdravlja prilikom napada neprijatelja, igrač uči da treba da ga izbegava i/ili da se bori protiv njega.

8. SISTEMI IGRE

U ovom odeljku biće opisani svi bitni sistemi i elementi igre, uključujući teren, inventar, krafter, napitke, sastojke i audio sistem.

8.1. Teren

Unity Terrain je alat u Unity-ju koji omogućava kreiranje i uređivanje velikih, realistično oblikovanih pejzaža za 3D igre [2]. U igri *Potion Master: Free the Ghosts* korišćen je za oblikovanje terena, dodavanje brda, dolina, trave i drveća. Koristili su se gotovi asseti [3], a vegetacija je animirana da izgleda kao da se pomera pod uticajem vetra.

U svetu igre postoji nekoliko jasno definisanih oblasti. Crvena šuma dominira drvećem crvene i narandžaste boje, sa ponekim neprijateljem koji ne predstavlja veliki izazov. Mračna šuma je najizazovnije mesto sa najvećim brojem neprijatelja i tmurnim izgledom. Polje cveća je miran deo igre, ukrašen cvećem i pun resursa, sa vrlo malo neprijatelja. Čarobničina koliba je glavno mesto u igri, odakle započinje avantura. Čarobnica mora da dođe do kolibe da bi napravila napitke, a ovo područje je bez neprijatelja.

8.2. Sastojci

Sastojci predstavljaju elemente igre koje igrač može sakupiti i koristiti za izradu napitaka. U svetu igre, sastojci se vizuelno ističu tako što se vrte oko svoje ose, čime se igraču signalizira da su interaktivni. Kada igrač stupi u kontakt s njima, sastojak nestaje iz igre, potvrđujući da je prikupljen.

8.3. Napici

U igri *Potion Master: Free the Ghosts* igrači mogu praviti razne napitke, koji su podeljeni u četiri osnovne grupe: napici za oslobađanje duhova, napici za borbu, napici za lečenje i napici za unapređenje. Tokom igre, igrači imaju uvid u recepte koji prikazuju potrebne sastojke za izradu napitaka. Da bi kreirali određeni napitak, igrači moraju imati sve potrebne sastojke u svom inventaru, a zatim otići do kraftera, koji je predstavljen kao čarobničin kazan, gde mogu napraviti napitak koji se automatski dodaje u inventar.

8.4. Inventar

Inventar je jedan od ključnih elemenata u igri *Potion Master: Free the Ghosts*, jer omogućava igračima da sakupljaju i upravljaju resursima koji su neophodni za napredovanje kroz igru. Ovaj sistem je osmišljen tako da bude jednostavan i intuitivan, što je posebno važno za mlađu publiku kojoj je igra namenjena.

Igrači pristupaju inventaru pritiskom na dugme za inventar, čime se otvara panel na kojem su prikazani svi resursi koje igrač poseduje, uključujući sastojke i napitke, prikazano na slici 4. Svaki predmet je jasno označen, a prikazane su i njihove količine, omogućavajući igračima da brzo uoče šta imaju na raspolaganju. Prilikom prikupljanja sastojaka, oni se automatski dodaju u inventar.

8.5. Krafter

Krafter predstavlja sistem za izradu napitaka. On je vizuelno predstavljen u obliku čarobničinog kazana, što je u skladu s tematikom igre. Kada igrač klikne na kazan, otvara se panel sa svim dostupnim napicima, na kome su prikazana imena napitaka i njihove količine, prikazano na slici 4. Ukoliko pored napitka stoji broj veći od nule, to znači da igrač može napraviti taj napitak.

Proces pravljenja napitaka je jednostavan. Igrač bira napitak koji želi da napravi, a sistem zatim automatski ažurira stanje sastojaka. Ovaj mehanizam pruža pregled dostupnih napitaka i sastojaka, što olakšava igraču upravljanje resursima i planiranje izrade. Krafter kao sistem osigurava da igrači mogu lako da vide koje napitke mogu napraviti.

8.6. Audio

Zvuk je veoma bitan u igrama. Obogaćuje iskustvo igranja i doprinosi atmosferi. Pozadinska muzika u igri *Potion Master: Free the Ghosts* pruža osećaj istraživanja stvarnog magičnog sveta.

Zvučni efekti doprinose većoj imerzivnosti igre. Zvučni efekat pri prikupljanju sastojaka govori igraču da je uspešno sakupio određeni resurs. Ukoliko se neprijatelj približi, čuje se zvuk pretnje, kako bi igrača upozorio na opasnost. Napadi neprijatelja su takođe propratni zvučnim efektima, čime igrač lakše izbegava udarce i reaguje na situaciju u igri.

9. KORISNIČKI INTERFEJS

Igra prati avanture mlade čarobnice, fokusirajući se na magiju i duhove, dok je njen vizuelni stil prilagođen prošlosti, ali i dalje šaren i privlačan. Korisnički interfejs je dizajniran da se uklopi u ovu estetiku. Kada igrač pokrene igru, prvo se susreće sa početnim menijem, iz kojeg može započeti igru, prilagoditi zvuk ili napustiti igru, slika 5.

Slika 4. *Inventar i krafter*

Ukoliko igrač započne igru, učitava se scena. Na ekranu se pojavljuje čarobnica iz trećeg lica, a u gornjem srednjem delu i mentor Grinbel koji usmerava igrača ka zarobljenom duhu. U gornjem levom uglu se nalazi health bar, dok donji deo ekrana sadrži kontrole za kretanje, napad i inventar.

Slika 5. *Početni meni*

Prilikom napada, igrač može baciti napitak, dok se otvorenim inventarom prati stanje resursa. Kroz interakciju s kazanom pored čarobnicine kuće, igrač može praviti napitke, a stanja sastojaka se ažuriraju. Ako igrač izgubi zdravlje, teleportuje se nazad do kolibe, gde mu se zdravlje vraća na maksimum, uz prikaz panela koji informiše o gubitku zdravlja i daje korisne savete.

Slika 6. *Početak igre*

10. ZAKLJUČAK

Ovaj rad je predstavio proces razvoja *Potion Master: Free the Ghosts*, akciono-avanturističke igre razvijene za mobilne telefone, namenjene deci avanturistima i ljubiteljima magije. Ispričana je šira priča igre. Predstavljeni su likovi, uključujući protagonistu čarobnicu, mentora Grinbela i antagonistu pauka. Objašnjen je i detaljan razvoj karaktera protagoniste, kao i gejmplej, mehanizmi igre, dizajn nivoa, najbitniji sistemi igre i korisnički interfejs.

Iako je igra gotova, rad prepoznaje mogućnosti za dalji razvoj. Dodavanje napitaka povećava mogućnosti gejmpleja. Dodavanje novih oblasti povećava mogućnosti igre i obogaćuje iskustvo igranja.

11. LITERATURA

[1] How did mobile games become so popular, Best Mobile App Awards bestmobileappawards.com/blog/how-did-mobile-games-become-so-popular (pristupljeno u septembru 2024.)

[2] Unity Technologies, „Unity – Terrain“ docs.unity3d.com/Manual/script-Terrain.html, Dokumentacija Unity Terrain. (pristupljeno u septembru 2024.)

[3] Modeli drveca i trave, Unity Asset Store, „Lowpoly Environment - Nature Free - MEDIEVAL FANTASY SERIES”, Polytope Studio assetstore.unity.com/packages/3d/environments/lowpoly-environment-nature-free-medieval-fantasy-series-187052 (pristupljeno u septembru 2024.)

Kratka biografija:

Bojana Karanović rođena je u Novom Sadu 2000. Diplomirala je na Fakultetu tehničkih nauka 2023. godine. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva odbranila je 2024. godine

kontakt: bojana.karanovic2015@uns.ac.rs

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.4
DOI: https://doi.org/10.24867/31BE13Jovicevic

ANALIZA PRETNjI I AUDITI U BLOCKCHAIN SISTEMIMA

THREAT ANALYSIS AND AUDITS IN BLOCKCHAIN SYSTEMS

Nikola Jovićević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIČKO I RAČUNARSKO INŽENjERSTVO

Kratak sadržaj – *U radu su opisani osnovni koncepti blockchain tehnologije i procesa bezbednosnih audita, uključujući metodologije, izazove i specifične karakteristike. Zatim je predstavljena analiza pretnji koje ugrožavaju sigurnost blockchain sistema, kao i različite strategije za njihovu identifikaciju i ublažavanje. Date sz preporuke za efikasnije sprovođenje audita u blockchain okruženju, sa ciljem unapređenja ukupne sigurnosti i pouzdanosti sistema.*

Ključne reči: *blokčejn, distribuirani sistemi, auditi, bezbednost, kriptografija*

Abstract – *The paper describes the fundamental concepts of blockchain technology and the process of security audits, including methodologies, challenges, and specific characteristics. It then presents an analysis of threats to blockchain system security, as well as various strategies for their identification and mitigation. Recommendations are provided for more effective implementation of audits in a blockchain environment, with the aim of improving overall system security and reliability.*

Keywords: *blockchain, distributed systems, audits, security, cryptography*

1. UVOD

Blockchain tehnologija prvi put je u današnjoj formi predstavljena kada je neidentifikovani pojedinac ili grupa, sa pseudonimmom Satoshi Nakamoto, objavio dokument "Bitcoin: A Peer-to-Peer Electronic Cash System"[1]. U tom radu predstavljena je matematička osnova na kojoj funkcioniše bitcoin. Inicijalna primena ove tehnologije ogledala se u uvođenju decentralizacije u finansijski sistem, a koja bi dalje omogućila sprovođenje transakcija bez posredovanja nekog od centralizovanih autoriteta, dozvoljavajući ljudima da upravljaju svojim novcem po sopstvenoj volji, uz sopstvenu odgovornost.

Obzirom na svoju upotrebu i potencijal, ovakvi sistemi se smatraju jako kritičnim kada je u pitanju sigurnost njihovog funkcionisanja, jer bi i najmanji propust mogao jako puno koštati sisteme, ali i njihove korisnike. Veliki značaj u tome svakako imaju sigurnosni auditi, revizije projekata namenjene pronalaženju propusta nastalih tokom kreiranja dizajna i implementacije kako bi se na vreme otkrile i otklonile potencijalne pretnje.

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

Naravno, nemoguće je garantovati da je neki proizvod apsolutno bezbedan, stoga se u verifikaciji koriste različite tehnike kako bi se suzbila mogućnost za nastanak greške. Najsigurnija je detaljna analiza koda i protokola, odrađena od strane više ljudi i timova, mada postoje i različiti alati za statičku analizu koji su u stanju da prepoznaju neke od ustaljenih greški.

2. AUDITI

Proces auditovanja, odnosno sprovođenja audita, u generalnom pogledu može se definisati kao verifikacija ili pregled kvaliteta nekog procesa ili sistema, kako bi se osigurala njegova usklađenost za zahtevima[5]. Audit se može odnositi na celokupnu organizaciju, a može biti ograničen i na određenu funkcionalnost, proces ili korak u proizvodnji. Pored toga mogu se odnositi i na administraciju, gde uključuju revizije dokumenata, rizika, kao i praćenje sprovedenih korektivnih akcija.

U pogledu informacionih sistema, auditi imaju višestruku ulogu i obuhvataju sve aspekte provere bezbednosti, od fizičke konfiguracije i okruženja u kome će sistem funkcionisati, preko softverske arhitekture, implementacionih detalja, rukovanja informacijama, pa sve do načina kako se koristi od strane korisnika. Njima se vrši procena koliko dobro sistem ispunjava zadate kriterijume, što dalje omogućava pronalazak i rešavanje problema pre nego što sistem bude pušten u rad, a u čemu se uviđa njihova ključna uloga u osiguravanju bezbednosti i pouzdanosti krajnjeg proizvoda.

3. BLOCKCHAIN

3.1. Distribuirani sistemi

Distribuirani sistemi predstavljaju sisteme sačinjene od više samostalnih računara koji su geografski udaljeni, koordinisani i komuniciraju preko mreže delujući međusobno u postizanju zajedničkog cilja. Programi koji se na njima izvršavaju dele se na segmente koji se izvršavaju na različitim mašinama. Nastali su kako bi se prevazišli različiti problemi tradicionalne klijent – server arhitekture, konkretnije, kako bi se poboljšale performanse, skalabilnost i dostupnost kroz raspodelu obrade podataka i zadataka između više čvorova.

3.2. Koncept distribuiranog lanca blokova

U distribuiranom sistemu sa mnogo nezavisnih čvorova, očekuje se postojanje zlonamernih aktera, pa se međusobno poverenje isključuje[2]. Svaka pristigla informacija se proverava pre postizanja konsenzusa, koristeći kriptografske hash funkcije. Svaki čvor čuva

svoju kopiju lanca blokova, što omogućava visoku redundantnost i sprečava manipulaciju podacima. Blokovi se dodaju putem konsenzusa, gde većina čvorova mora potvrditi validnost novog bloka.

3.2.1. Struktura bloka

Blokovi najčešće sadrže zaglavlje sa verzijom protokola, hash prethodnog bloka, Merkle root, broj bloka, vremensku oznaku, ciljne vrednosti za rudarenje, i nonce. Telo bloka sadrži broj i listu transakcija[1].

3.2.2. Tok transakcije i konsenzus

Postizanje konsenzusa može se ostvariti raznim metodama, najčešće PoW (Proof of Work) i PoS (Proof of Stake). Transakcije se propagiraju kroz mrežu, a jedan čvor predlaže blok koji ostali čvorovi validiraju. U PoW sistemima se koristi *nonce* za izbor validatora koji predlaže blok, dok PoS sistemi biraju validatore na osnovu različitih faktora, najčešće na osnovu uloženih, odnosno delegiranih sredstava, ali i nasumičnim izborom.

Decentralizovani sistemi, sa druge strane, raspoređuju kontrolu na više nezavisnih čvorova, koji funkcionišu bez centralnog autoriteta. Oni nude prednosti kao što su otpornost na greške, otpornost na napade i skalabilnost, ali istovremeno uvode složenosti u koordinaciju i komunikaciju među čvorovima, kao i potrebu za algoritmima za sprovođenje konsenzusa.

3.4. Ograničenja i pravci unapređenja blockchain tehnologije

Kao i svaka druga tehnologija, blockchain ima svoja ograničenja u primeni, ali je sklon i evoluciji, promenama i unapređenjima. Brojni su razlozi zašto ne bi trebalo žuriti sa primenom ove tehnologije u svim sferama u kojima je ona primenjiva, a primarni je zato što je ona još uvek mlada i treba sačekati da se dodatno razvije. To će, naravno, zahtevati još vremena uloženog u razvoj tehnologija i algoritama koji će vremenom učvrstiti i verovatno proširiti spekar primene *blockchain*-a. Neki od problema sa kojima se ova grana industrije trenutno susreće i principi za njihovo prevazilaženje su[3]:

- **Nedostatak privatnosti**: Svi čvorovi održavaju celokupnu istoriju transakcija, što smanjuje privatnost. Rešenja poput ZK-SNARK, ZK-STARK, i L2 rešenja (npr. Lightning Network za Bitcoin i zk-Rollups za Ethereum) omogućavaju verifikaciju transakcija bez otkrivanja specifičnih podataka.
- **Visoki troškovi procesiranja transakcija**: Ovi troškovi štite od manipulacija, ali mogu povećati operativne troškove. *Layer 2* rešenja kao što su *rollup*-i i *sharding* smanjuju opterećenje glavnog lanca i poboljšavaju skalabilnost.
- **Bezbednosni modeli i gubitak sredstava**: *Blockchain* se oslanja na javne i privatne ključeve, bez rešenja za slučaj gubitka privatnog ključa. Korišćenje hardverskih i *multisig* novčanika smanjuje rizik od gubitka i zloupotrebe.
- **Kašnjenje**: Postizanje konsenzusa u distribuiranoj mreži može biti sporo. *Latency* se može smanjiti optimizacijom mrežnih protokola, korišćenjem PoS i DPoS pristupa, smanjenjem veličine blokova, *sharding*-om, i L2 rešenjima.

Blockchain sistemi takođe evoluiraju prema interoperabilnosti, omogućavajući komunikaciju između različitih *blockchain*-ova putem višelančanih protokola i mostova (*bridges*), kao što su Polkadot i Cosmos sa IBC protokolom.

4. AUDITI U BLOCKCHAIN-U

Blockchain tehnologija je vremenom ostvarila raznovrsnu primenu, ali pored svih benefita, rad u ovom okruženju sa sobom nosi i visoke rizike. S jedne strane je činjenica da je uglavnom u pitanju rad sa novcem, gde tačnost i bezbednost podataka ne smeju biti dovedeni u pitanje, dok je sa druge strane perspektiva i poverenje korisnika, čiji bi gubitak nepovoljno uticao na motivaciju za dalji razvoj. Ova kritičnost upravo naglašava potrebu za temeljnim revizijama projekata, koji se u celokupnom ekosistemu nameću kao ključni za identifikaciju potencijalnih pretnji i ranjivosti u sistemu, kao i generalnu validaciju projekata. Povećana odgovornost koju auditori preuzimaju potiče iz potrebe da se obezbedi integritet i sigurnost sistema koji radi sa finansijskim sredstvima i osetljivim podacima. Kako bi se to izvelo, svi učesnici u procesu moraju poslovati odgovorno.

4.1. Audit request dokument

Pri definisanju specifikacije audita, neophodno je jasno, jednoznačno i precizno definisati očekivanja, ali i osigurati da je relevantnim ljudima obezbeđen pristup do svih neophodnih izvora. Za ove svrhe, pre početka projekta, popunjava se zahtev za sprovođenje audita (*Audit Request*). On može predstavljati formu ili dokument kroz koji će klijent, odnosno softverska kuća koja zahteva audit, popuniti sa ciljem da se kroz kratak opis i obezbeđene resurse predstavi projekat, kako bi se mogla sprovesti precizna procena vremena i ljudstva potrebnog da se revizija sprovede. Dobro pripremljen zahtev za audit treba da sadrži:

1. Precizno definisan opseg audita (*scope*) – navesti delove sistema koji će biti pregledani, uključujući repozitorijume, module, biblioteke;
2. Resurse – lista relevantnih izvora (dokumentacija, kod, dijagrami) kako bi se audit tim bolje pripremio;
3. Željeno vreme do kada bi audit trebalo završiti (timeline) – razuman vremenski okvir na osnovu složenosti projekta;
4. Očekivanja od audita – definiše specifične zahteve i usluge;
5. Očekivani rezultat (output) – definiše šta će biti obezbeđeno na kraju audita (npr. izveštaj, issue-i na GitHub-u);
6. Relevantne kontakte sa obe strane – lista relevantnih osoba za komunikaciju između klijenata i auditora;

7. Dodatne napomene firme koja sprovodi audit – napomene o pripremi koda i resursa pre slanja zahteva.

Nakon razmatranja svih parametara, vrši se procena potrebnog vremena i resursa, a zatim se klijentu predstavlja procena i organizuje početni sastanak *(kick-off meeting*), gde će se razgovarati o detaljima neophodnim za početak posla. Nakon toga organizuju se sastanci po dogovoru, koji se sprovode jednom ili više puta nedeljno, a gde se diskutuje o projektu i potencijalno pronađenim problemima, sve do zaključivanja projekta i predstavljanja finalnog izveštaja o sprovedenoj reviziji na završnom (*closure*) sastanku.

4.2. Očekivanja od audita

Jasno je da sprovođenje audita nije trivijalan proces, jer se ni audit sam po sebi ne svodi samo na analiziranje koda i traženje grešaka. Različite kompanije mogu koristiti različite tehnike i alate u pronalaženju problema, pa se stoga klijentima odmah treba skrenuti pažnja kakve usluge mogu da očekuju, i da u skladu sa tim odaberu šta im je od interesa. Ono što uglavnom spada u osnovne usluge audita su:

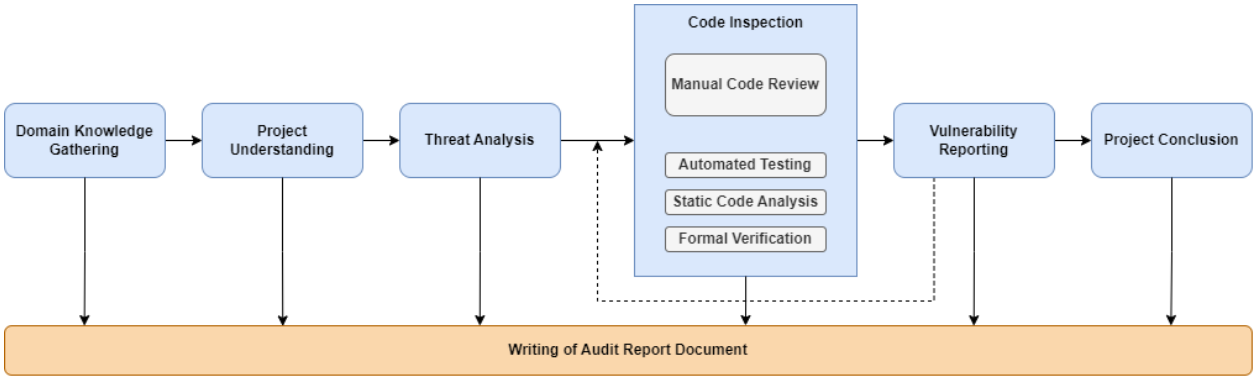
1. analiza specifikacije projekta;
2. pregled dokumentacije zbog potencijalnih nejasnoća i nekonzistentnosti sa kodom;
3. analiza koda i funkcionalnosti;
4. pregled ispravki problema prijavljenih u okviru audita.

Pored toga, neki alternativni pristupi uključuju i analizu arhitekture i protokola koja može obuhvatiti i sugestije za njihovo unapređenje, analizu i unapređenje testova, ali i pomoć pri odlukama u procesu dizajniranja rešenja za određene probleme, ili funkcionalnosti koje će tek biti implementirane. Tu se takođe mogu vršiti i neke od sofisticiranijih metoda, poput formalne verifikacije softvera, pisanjem matematičkih specifikacija upotrebom jezika kao što su TLA+ i Quint.

4.3. Tehničke složenosti i izazovi

Iako suštinski mogu biti nepredvidivi, auditi se mogu grubo podeliti na nekoliko faza (slika 1), gde za svaku treba izdvojiti dovoljno vremena kako bi dalji tok projekta mogao biti što bolje sproveden.

Prva faza nastupa pre početka same analize, gde se nakon utvrđivanja zahteva pristupa istraživanju relevantnih oblasti i tehnologija (izučavanje domena).



Slika 1. *Koncepti i faze audita*

Nakon nje započinje izučavanje konkretnog projekta i upoznavanje sa potrebnim segmentima sistema. U ovoj fazi veliku ulogu igraju projektne dokumentacije, slike, dijagrami, video prezentacije i sl. U slučaju nedostatka gorepomenutih izvora, pristupa se procesu reverzivnog inženjeringa.

Zatim se razmatraju potencijalni pravci napada, ustaljene pretnje relevantne za projekte koji imaju slične karakteristike, koncepte i namenu. Sumiraju se i flitriraju ranija iskustva kako bi se proverilo prisustvo ili nedostatak odgovarajućih mehanizama i algoritama koji bi ciljane pretnje adekvatno adresirali.

Analiza koda sprovodi se iterativno, prolazeći kroz iste segmente koda više puta od strane više auditora. Razlikujemo analizu koda „liniju po liniju“ i analizu upotrebom različitih alata za statičku analizu.

4.4. Modelovanje pretnji

Kako bi se sprovela detaljna analiza pretnji i razmotrili potencijalni vektori napada, neophodno je za analizirani sistem kreirati adekvatan model pretnji (*threat model*), koji bi trebalo da opisuje osnovne karakteristike pojedinih delova sistema ili korišćenih algoritama koji bi potencijalno mogli uvesti slabosti u sistem. Zbog ovoga se u obzir mora uzeti mnoštvo aspekata, poput učesnika i komponenti u sistemu, arhitekturu, tokove podataka, protokole za komunikaciju i konsenzus, njihova ograničenja i slučajeve upotrebe u odnosu na posmatrano okruženje i zahteve koje bi trebalo da ispunjavaju.

4.5. Raspodela i vektori napada

Analizom različite literature koja se bavila analizom napada na *blockchain* sisteme sa ograničenim pristupom, identifikovani su napadi i komponente koje su im podložne[4].

Neke od najučestalijih pravaca napada uključuju zloupotrebu ranjivosti pametnih ugovora, propusta u implementaciji matematičkih operacija i problema uzrokovanih pojavama *overflow*-a i *underflow*-a.

Velike probeleme mogu uzrokovati i kriptografske ranjivosti, koje ukoliko postoje, predstavljaju izuzetno ozbiljnu pretnju budući da integritet i neporecivost samog

blockchain-a zavisi od ispravnosti *hash* funkcija i digitalnih potpisa.

Kada su u pitanju *bridge* lanci namenjeni *cross-chain* komunikaciji i prenosu sredstava između različitih *blockchain* mreža, karakteristični su *replay* napadi, gde se koristi već izvršena, validna transakcija sa jedne mreže na drugu, pri čemu ona još uvek nije potpuno procesirana i evidentirana kao iskorišćena, uzrokujući *double minting*, odnosno *double burning*.

Prepoznatljivi DoS napadi, iako nisu previše zastupljeni, u *blockchain* okruženju ipak predstavljaju pretnju. Kako bi napadač izveo uspešan DoS napad, može pokušati da preplavi neke od čvorova karakterističnim zahtevima koje validator treba obraditi (npr. TCP SYN paketima). Ukoliko je napad usmeren na trenutnog lidera, može u značajnoj meri smanjiti ili potpuno obustaviti proces postizanja konsenzusa.

Segmentacija mreže (*network partitioning*), može se desiti ako se u sistemu pronađu zlonamerni validatori koji manipulacijom konsenzus protokola putem protokola na mrežnom nivou mogu particionisati mrežu koristeći napade poput BGP *hijacking*-a i DNS napada. Nakon što je mreža podeljena, sledi manipulacija procesom donošenja odluka, koja može uključivati stvaranje falsifikovanih blokova koji bi potom uticali na validnost i integritet lanca. Ovo se može sprovesti generisanjem lažnih čvorova ili izolovanjem određenog čvora u mreži koji bi video samo one podatke koje napadać želi da prikazuje, time omogućavajući manipulaciju podacima koje konkretan čvor vidi i šalje.

4.6. Audit Report dokument

Audit *report* dokument, predstavlja finalni izveštaj sa audit projekta koji obuhvata evidenciju o celokupnom poslu koji je urađen. Svaki dokument se razlikuje i zavisi od prakse audit kuće koja je autor. Centralni deo koji je svakako najvažniji predstavlja popis pronađenih problema i preporuka za njihovo rešavanje, ali on može sadržati i mnogo više, što je svakako dobra praksa. Sveobuhvatni izveštaj sa audita koji na transparentan način predstavlja i tok samog audita sadrži poglavlja za gotovo svaku fazu i uključuje:

1. Kratak pregled projekta (*Project Overview*);
2. Tehničke detalje vezane za audit (*Audit Dashboard*);
3. Analiza sistema (*System Overview*);
4. Analiza pretnji i invarijanti (Threat / Invariant Analysis);
5. Lista i opis pronađenih propusta (*Findings*);
6. Klasifikacija *finding*-a (*Findings Classification)*;
7. Odricanje od odgovornosti i dodatne napomene (Disclaimer).

5. ZAKLjUČAK

Potencijal *blockchain*-a postao je neupitan, i jedino se postavlja pitanje dokle će sezati granice njegove primene. Pred ovim konceptom ipak stoji još mnogo izazova, godina istraživanja i razvoja, a u svemu tome biće neophodna podrška u revizijama, verifikacijama kvaliteta i savetima za pravce unapređenja. Uloga audita u ovom celokupnom podvigu nije mala jer su upravo audit inženjeri oni koji dolaze u dodir sa raznovrsnim rešenjima i pristupima, i stiču široku sliku o tehnologijama koje se razvijaju. Iz tog razloga je ovaj poziv karakterističan, jer zahteva konstantno učenje novih tehnologija u kratkom vremenskom rasponu i doprinosi sticanju iskustva u mnogim domenima.

U okviru ovog rada, sumirani su ključni izazovi i specifične karakteristike audita u kontekstu *blockchain* tehnologije, kao i strategije za identifikaciju i ublažavanje pretnji koje ugrožavaju sigurnost *blockchain* sistema. Ovo je značajno jer pruža bolji uvid u kompleksnost bezbednosnih audita i nudi smernice za njihovo efikasno sprovođenje. Kroz analizu postojećih pretnji i potencijalnih napada, osvetljavaju se segmenti tehnologije u kojima je potrebno dodatno istraživanje, ali i kontinuirano unapređenje postojećih metodologija.

U svetlu sve brže evolucije blockchain tehnologije neophodno je nastaviti sa istraživanjem i razvojem novih pristupa u sprovođenju audita, kako bi nastavili sa obezbeđivanjem sigurnosti i pouzdanosti sistema. Ova istraživanja mogu dalje ići u različitim pravcima, baviti se analizama novih pretnji, razvojem automatizovanih alata za detekciju ranjivosti, ili unapređenjem procedura audita u skladu sa specifičnostima novonastalih protokola.

6. LITERATURA

[1] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." *Satoshi Nakamoto* (2008).

[2] Chaum, David Lee. *Computer systems established, maintained and trusted by mutually suspicious groups*. Diss. University of California, Berkeley, 1982.

[3] Hughes, Laurie, et al. "Blockchain research, practice and policy: Applications, benefits, limitations, emerging research themes and research agenda." *International journal of information management* 49 (2019).

[4] Putz, Benedikt, and Günther Pernul. "Detecting blockchain security threats." *2020 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 2020.

[5] https://asq.org/quality-resources/auditing (pristupljeno u septembru 2024)

Kratka biografija:

Nikola Jovićević rođen je u Užicu 1999. godine. Odrastao je u Požegi gde je završio osnovnu školu i gimnaziju. Osnovne akademske studije na Fakultetu tehničkih nauka Univerziteta u Novom Sadu upisao je 2018. godine. Diplomirao je u septembru 2022, i iste godine upisao master akademske studije.
kontakt: n.jovicevic999@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.39
DOI: https://doi.org/10.24867/31BE14Jankov

PREPOZNAVANJE KARAKTERISTIKA MODULACIJE PRIMENOM TEHNIKA DUBOKOG UČENJA

AUTOMATIC MODULATION RECOGNITION USING DEEP LEARNING TECHNIQUES

Milica Jankov, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIČKO I RAČUNARSKO INŽENJERSTVO

Kratak sadržaj – *U ovom radu analiziraju se performanse dva modela mašinskog učenja u prepoznavanju modulacionih šema: konvoluciona (CNN) i LSTM neuronska mreža. Koristeći sintetički skup podataka RadioML2016.10a, modeli su obučavani i testirani na 11 različitih tipova modulacija u opsegu SNR od -20 dB do +20 dB. Rezultati pokazuju da oba modela ostvaruju visoku tačnost, s tim da LSTM mreža pokazuje blagu prednost u performansama prilikom klasifikacije modulacija pri višim SNR vrednostima. Ova analiza doprinosi razvoju naprednih tehnika prepoznavanja modulacije koje mogu poboljšati efikasnost kognitivnog radija u složenim spektralnim okruženjima.*

Ključne reči: *Automatsko prepoznavanje modulacije, Kognitivni radio, Konvolucione neuronske mreže, Long Short-Term Memory, Duboko učenje*

Abstract – *This paper analyzes the performance of two machine learning models in modulation scheme recognition: Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) network. Using the synthetic RadioML2016.10a dataset, the models were trained and tested on 11 different modulation types within an SNR range from -20 dB to +20 dB. The results show that both models achieve high accuracy, with the LSTM network demonstrating slightly better performance in modulation classification at higher SNR values. This analysis contributes to the development of advanced modulation recognition techniques that can enhance the efficiency of cognitive radio in complex spectral environments.*

Keywords: *Automatic Modulation Recognition, Cognitive Radio, Convolutional Neural Networks, Long Short-Term Memory, Deep Learning*

1. UVOD

Globalni porast umreženih uređaja stvara veliku potražnju za radio-frekvencijskim spektrom, što čini tradicionalni pristup dodeljivanja spektra, gde su fiksni frekvencijski opsezi ekskluzivno dodeljeni licenciranim primarnim korisnicima (*Primary Users*, PU), neodrživim.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio doc. dr Milan Narandžić.

Tehnologija kognitivnog radija rešava problem neiskorišćenog spektra, tako što omogućava nelicenciranim sekundarnim korisnicima (*Secondary Users*, SU) da koriste neaktivne kanale primarnih korisnika bez izazivanja smetnji. Sekundarni korisnici pri tome moraju primeniti robustne tehnike detekcije spektra (*Spectrum Sensing*, SS) kako bi precizno detektovali prisustvo ili odsustvo primarnih korisnika u kanalima.

Automatsko prepoznavanje modulacije (*Automatic Modulation Recognition*, AMR) predstavlja jedan od ključnih segmenata razvoja kognitivnog radija (*Cognitive Radio*, CR), koji omogućuje adaptivnu demodulaciju detektovanih signala [1]. Pretpostavka o poznavanju tipa modulacije omogućava sekundarnim korisnicima da preciznije razlikuju različite primarne korisnike, čime dobijaju detaljniji uvid o spektru, što im pomaže u donošenju informisanih odluka o tome kada i gde treba da obave prenos. Rezultati prepoznavanja modulacije mogu dodatno doprineti identifikaciji karakteristika interferirajućih signala, omogućavajući primenu ciljanih tehnika za suzbijanje interferencije. U cilju postizanja što tačnijih rezultata, ovaj rad se fokusira na analizu performansi različitih modela namenjenih rešavanju problema klasifikacije modulacionih šema.

2. PREPOZNAVANJE MODULACIJE

Osnovni cilj ispitivanja zauzetosti spektra (*spectrum sensing*) je da omogući saznanja o prisutnim predajnicima kako bi se izbegla radio interferencija i optimizovala raspodela spektra. To podrazumeva identifikaciju emitovanih radio signala i drugih potencijalnih smetnji, pri čemu sve emisije poseduju određene specifičnosti. U tom kontekstu, prepoznavanje modulacije ima za cilj klasifikaciju tipa modulacije na osnovu primljenog radio signala, što predstavlja važan korak ka identifikaciji komunikacionih šema i predajnika u okolini.

2.1. Potreba za primenom mašinskog učenja

Prepoznavanje modulacije može se tretirati kao problem klasifikacije u više klasa, gde se odluka donosi za jednu od N klasa. Postojeće metode tog tipa mogu se podeliti u dve grupe: metode zasnovane na verodostojnosti (*likelihood-based*) i metode zasnovane na obeležjima (*feature-based*) [2]. Kao alternativa navedenim pristupima, nedavno su se pojavile metode zasnovane na dubokom učenju, koje direktno uče iz primljenih signala. Cilj je ispitati da li predložene tehnike pružaju veću fleksibilnost u učenju ključnih karakteristika u poređenju s tradicionalnim,

analitičkim pristupima koji se oslanjaju na ljudsku ekspertizu. Za poređenje ostvarenih performansi biće korišćena tačnost klasifikacije.

2.2. Formulacija problema

Ulazni podatak predstavlja primljeni signal u obliku kompleksnog vremenskog niza u osnovnom opsegu. Uzorkovanje komponenti radio signala u fazi i kvadraturi obavlja se u diskretnim vremenskim intervalima, što rezultuje kompleksnim vektorom veličine $1 \times N$.

Takav zapis predstavljen je u jednačini (1),

$$r(t) = s(t) \cdot c + n(t), \quad (1)$$

gde s(t) označava vremenski niz signala, predstavljen u kontinualnom ili diskretnom vremenu. Ovaj signal se sastoji od niza bita modulisanih u sinusoidu sa promenljivim parametrima koji uključuju frekvenciju, fazu, amplitudu, putanju ili neku kombinaciju ovih faktora. Konstanta c označava slabljenje duž putanje (*path loss*) ili konstantno pojačanje signala, dok n(t) predstavlja aditivni beli Gausov proces koji označava termički šum.

Osnovni cilj bilo kog klasifikatora modulacije je da izrazi verovatnoću

$$P(s(t) \in N_i | r(t)), \quad (2)$$

gde je *r(t)* jedini referentni signal, a N_i predstavlja *i*-tu klasu. Primljeni signal *r(t)* se obično predstavlja u *IQ* formatu zbog svoje fleksibilnosti i jednostavnosti za primenu matematičkih operacija i dizajn hardvera. Komponente signala u fazi i kvadraturi izražene su kao:

$$I = A\cos(\varphi), \quad (3)$$

i

$$Q = A\sin(\varphi), \quad (4)$$

gde *A* i φ predstavljaju trenutnu amplitudu i fazu primljenog signala *r(t)*.

Za problem klasifikacije modulacije, *IQ* komponente uzorkovanog signala predstavljaju obeležja od interesa. Pored formatiranja ulaznih podataka u vidu *IQ* uzoraka (pravougaone koordinate), moguće je koristiti predstavljanje ulaznih podataka kroz amplitudu i fazu (polarne koordinate).

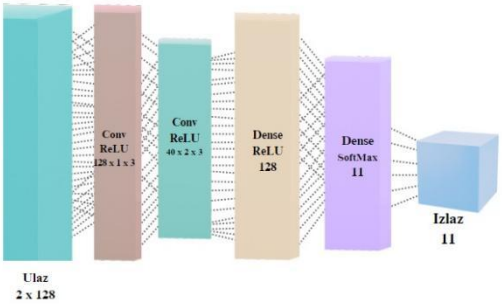
3. RAZMATRANE METODE ZA AUTOMATSKU DETEKCIJU KARAKTERISTIKA MODULACIJE

3.1. Konvoluciona neuronska mreža (CNN)

U sistemima radio komunikacija, često razmatrana klasa prijemnika uključuje korišćenje prilagođenog filtra. Na prijemnoj strani, za svaki specifično dizajnirani filtar, uparen sa odgovarajućim prenetim simbolom, vrši se konvolucija sa dolaznim vremenskim signalom. Motivacija za primenu konvolucionih neuronskih mreža (*Convolutional Neural Network*, CNN) u ovom kontekstu leži u očekivanju da će CNN moći automatski da oblikuju prilagođene filtre za različite vremenske karakteristike signala, dok istovremeno optimizuju performanse pri različitim nivoima odnosa SNR. Cilj je iskoristiti invarijantnost konvolucione neuronske mreže na pomeranje, kako bi se kreirali prilagođeni filtri sposobni da automatski izdvoje ključne karakteristike prenetih simbola, bez dodatnih mrežnih informacija ili estimacije osnovnog talasnog oblika. Ovakav pristup bi mogao da obezbedi robustnu osnovu za rešavanje problema klasifikacije modulacija.

Prvi metod za učenje obeležja, konvoluciona neuronska mreža (CNN), koristi prozorirani ulaz originalnog vremenskog niza radio signala r(t). Ovaj kompleksni signal tretira se kao dvodimenzionalni realni ulaz, gde se r(t) predstavlja kao 2 x N vektor u kompaktnoj 2D konvolucionoj mreži. Dimenziju veličine 2 formiraju ortogonalno sinhronizovani uzorci u fazi i kvadraturi (*IQ*).

CNN model koji se koristi u ovoj arhitekturi (slika 1) sastoji se od četiri sloja — dva konvoluciona sloja i dva potpuno povezana - gusta (*dense*) sloja. Nakon konvolucionih slojeva, izlaz se spaja sa prvim potpuno povezanim - gustim (*dense*) slojem, koji integriše informacije iz prethodnih slojeva i zatim ih prosleđuje ka završnom klasifikacionom sloju. Završni sloj je drugi potpuno povezani - gusti (*dense*) sloj koji sadrži n neurona, gde je n broj klasa u klasifikacionom zadatku (u razmatranom slučaju, n = 11). Obuka se sprovodi koristeći kategoričku unakrsnu entropiju (*categorical cross-entropy*) kao funkciju gubitka, dok je za optimizaciju odabran Adam (*Adaptive Moment Estimation*) optimizator [3].



Slika 1. *Ilustracija CNN arhitekture.*

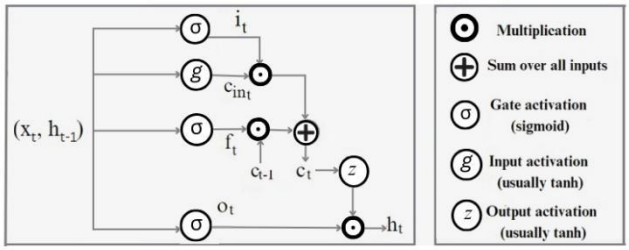
3.2. Long Short-Term Memory (LSTM)

Naredni metod za učenje obeležja koristi LSTM (*Long Short-Term Memory*) neuronske mreže. Za razliku od prethodno opisanog pristupa, ovaj model se obučava na osnovu informacija o amplitudi i fazi u vremenskom domenu, oslanjajući se na modulacione šeme iz podataka za obuku, bez potrebe za dodatnim ekspertskim obeležjima, poput cikličnih momenata višeg reda. Zahvaljujući LSTM ćelijama, koje su poznate po efikasnosti u učenju dugih vremenskih sekvenci, očekuje se da bi predloženi model mogao da prepozna dugoročne zavisnosti u podacima, bez potrebe za eksplicitnim izdvajanjem obeležja.

LSTM model koji se opisuje u nastavku predstavlja poseban tip Rekurentnih neuronskih mreža (*Recurrent Neural Networks*, RNN), često korišćenih za izdvajanje obeležja iz vremenskih nizova podataka. Blok dijagram osnovne verzije LSTM ćelije prikazan je na slici 2.

Osnovu funkcionisanja LSTM ćelije čini njeno interno stanje ili memorija c_t zajedno sa tri ulaza: ulaz za unos podataka (input gate, i_t), mehanizam za kontrolu pamćenja (forget gate, f_t) i krajnji izlaz (output gate, o_t). Na osnovu prethodnog stanja i ulaznih podataka, ćelije mogu naučiti

ulazne težine za određeni problem. Ovaj mehanizam koji je zasnovan na različitim ulazima, omogućava LSTM ćelijama da skladište informacije tokom dužeg vremenskog perioda, što podstiče učenje trajnih obeležja.



Slika 2. *LSTM ćelija korišćena u skrivenim slojevima modela.*

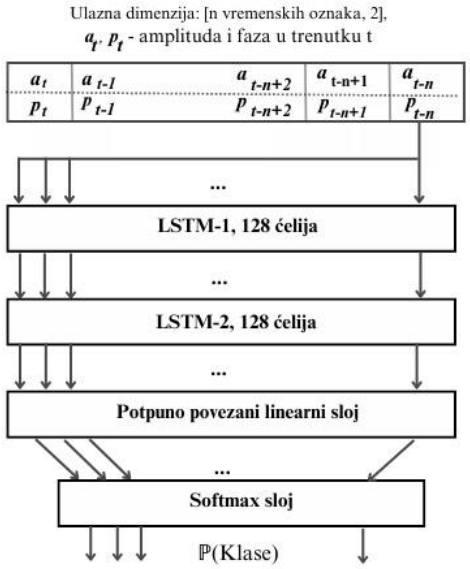
Za klasifikaciju kompleksnih signala koristi se LSTM mreža sa dva sloja, kao što je prikazano na slici 3. U svakom trenutku, t, vrednosti, amplituda i faza modulisanog signala, unose se u LSTM ćelije u obliku dvodimenzionalnog vektora. Vrednosti amplituda su normalizovane primenom L_2 norme, dok je faza, izražena u radijanima, skalirana na opseg od -1 do 1. Prva dva sloja modela sadrže po 128 LSTM ćelija. Konačni izlaz iz drugog LSTM sloja, vektor dimenzije 128, koristi se kao ulaz za završni gusti sloj (*dense layer*). Ovaj sloj je gusti Softmax sloj (*dense Softmax layer*) koji mapira naučena obeležja na jednu od 11 klasa, koje predstavljaju različite šeme modulacije. Aktivaciona funkcija koja se koristi u ovom sloju jeste Softmax, koja omogućava modelu da predvidi verovatnoće za svaku klasu, čime daje procenu klase kojoj signal pripada.

4. REZULTATI

Skup podataka koji je sintetički generisan korišćenjem GNU Radio [4], pod nazivom *RadioML2016.10a*, koristi se u ovom radu za obuku i procenu performansi klasifikatora. Ključna metrika uspešnosti bila je postignuta tačnost na test skupu. U postupku evaluacije, podaci su podeljeni na skup za obuku (60%) i skupove za validaciju i testiranje (po 20% svaki). Obuka je sprovedena na približno 33 miliona kompleksnih uzoraka, podeljenih u 11 različitih modulacionih šema, koje u nastavku označavamo standardnim akronimima: 8PSK, AM-DSB, AM-SSB, BPSK, CPFSK, GFSK, 4PAM, 16QAM, QPSK i WBFM. Raspoloživi uzorci su grupisani u segmente od po 128 instanci. Skupovi za validaciju i testiranje sadrže približno 11 miliona kompleksnih uzoraka, koji su ravnomerno raspoređeni prema različitim vrednostima odnosa signala i šuma (SNR), u rasponu od -20 dB do +20 dB. Ovo omogućava procenu performansi klasifikatora u prisustvu različitih nivoa šuma.

4.1. CNN

Na osnovu matrice konfuzije za CNN klasifikator pri odnosu SNR od 18 dB (slika 4), moguće je primetiti nekoliko važnih aspekata. Modulacije poput CPFSK, 4-PAM, BPSK, i GFSK imaju izuzetno veliku tačnost, dostižući vrednosti bliske 100%. To znači da CNN klasifikator veoma uspešno prepoznaje ove modulacije pri visokom odnosu SNR, gde je šum minimalan, što olakšava tačno razlikovanje karakteristika signala.



Slika 3. *Dvoslojni LSTM model za klasifikaciju*.

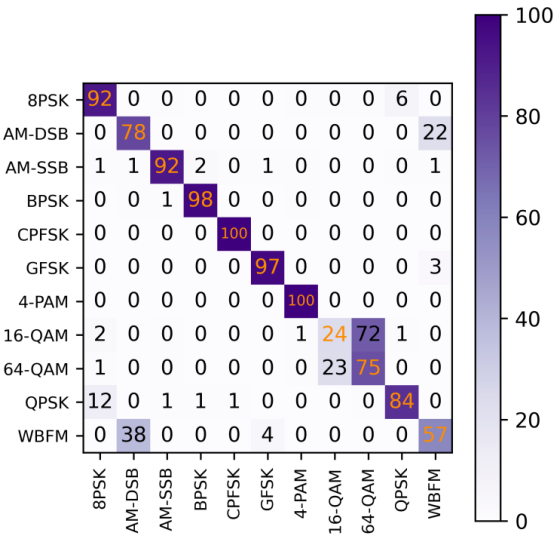
Pri razlikovanju 16-QAM i 64-QAM, CNN i dalje pokazuje određeni stepen međusobne zbunjenosti. Iako je tačnost za 64-QAM (75%) solidna, postoji značajan broj uzoraka iz 16-QAM (72%) koji su pogrešno klasifikovani kao 64-QAM, što ukazuje na to da CNN klasifikator nije sasvim uspešan u razdvajanju ova dva tipa modulacija čak ni pri visokom odnosu SNR. Potencijalni razlog za to jesu sličnosti u njihovim kvadraturnim amplitudskim osobinama, gde konstelacija nižeg reda predstavlja podskup tačaka iz konstelacije višeg reda.

Iako QPSK pokazuje solidnu tačnost (84%), postoji određeni nivo zbunjenosti sa 8PSK (12% uzoraka QPSK pogrešno klasifikovano kao 8PSK). Ovim se primećuje da fazne modulacije sa višestrukim faznim stanjima predstavljaju izazov za CNN, iako pri višim odnosima SNR klasifikacija postaje preciznija. WBFM pokazuje malu tačnost (57%) i čestu pogrešnu klasifikaciju kao AM-DSB, gde je 38% uzoraka WBFM signala pogrešno klasifikovano kao AM-DSB. U skladu sa prethodnim zapažanjima, moguće je zaključiti da CNN klasifikator ima problem sa razlikovanjem ove dve modulacije, verovatno zbog sličnih karakteristika njihovih osobina.

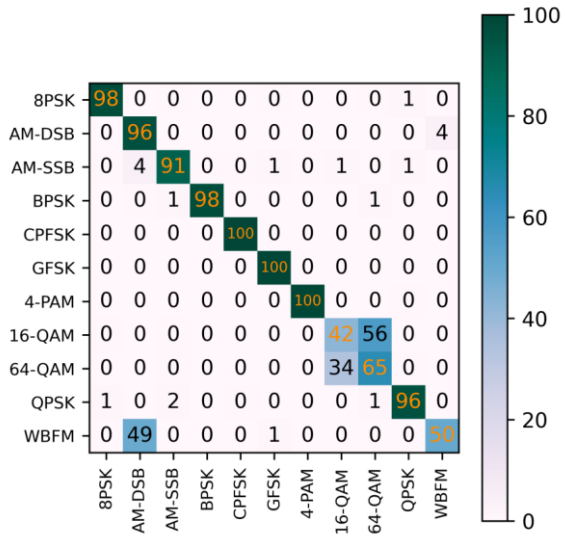
Dakle, pri odnosu SNR od 18 dB, CNN klasifikator pokazuje vrlo visoku tačnost (81.54%) za većinu modulacija, posebno za CPFSK, BPSK, i GFSK. Međutim, i dalje postoje problemi sa međusobnim razlikovanjem 16-QAM i 64-QAM, kao i značajnim neraspoznavanjem između WBFM i AM-DSB.

4.2. LSTM

Na osnovu prikazane matrice konfuzije za LSTM klasifikator pri SNR od 18 dB (slika 5), sa ukupnom tačnošću od 85.05%, moguće je uočiti nekoliko ključnih aspekata. Velika tačnost za određene modulacije 8PSK, AM-DSB, BPSK, CPFSK, GFSK, 4-PAM i QPSK (iznad 96%), ukazuje na to da LSTM model izuzetno dobro klasifikuje ove modulacije pri 18 dB. Na primer, za CPFSK, GFSK, i 4-PAM, tačnost je 100%, što znači da nijedan uzorak ovih modulacija nije pogrešno klasifikovan.



Slika 4. *Matrica konfuzije pri odnosu SNR od 18 dB za CNN klasifikator.*



Slika 5. *Matrica konfuzije pri odnosu SNR od 18 dB za CNN klasifikator.*

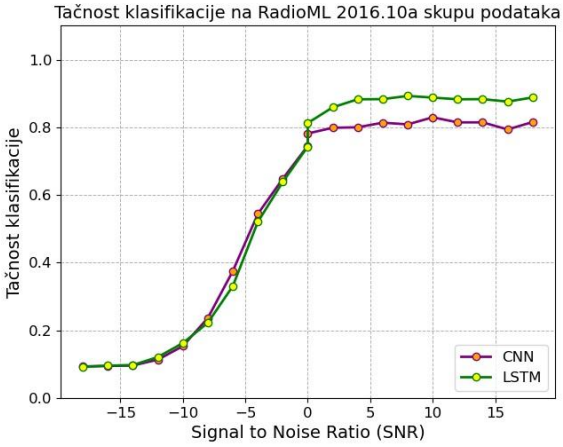
Postoji nekoliko manjih grešaka kod AM-SSB, ali generalno ova modulacija pokazuje visok nivo tačnosti (91%), što znači da je model uspešan u klasifikaciji ove modulacije pri većim vrednostima SNR. Model postiže vrlo dobre rezultate za većinu modulacija, ali ima određene poteškoće sa razlikovanjem sličnih kvadraturnih modulacija (16-QAM i 64-QAM) i sa WBFM modulacijom.

Slično kao u prethodnom eksperimentu, problemi prilikom klasifikovanja WBFM modulacije potvrđuju da ona predstavlja izazov za klasifikaciju čak i pri visokom odnosu SNR. Ostvarena tačnost iznosi 50%, a 49% slučajeva je greškom klasifikovano kao AM-DSB. Ova matrica konfuzije ukazuje na najznačajnije probleme u klasifikaciji, i može biti smernica za dalja poboljšanja modela.

5. ZAKLJUČAK

Sprovedeni eksperimenti pokazuju da drugi analizirani model, koji se zasniva na primeni LSTM ćelija, dovodi do povećanja tačnosti od 3.51% (na test skupu) u poređenju sa tačnošću koja se postiže primenom modela zasnovanog na konvolucionim neuronskim mrežama. Prosečna tačnost klasifikacije za SNR vrednosti u rasponu od 0 dB do 18 dB, postignuta primenom CNN i LSTM modela, iznosi 55.86% i 58.87%, respektivno. Prikaz na slici 6 ističe prednost LSTM modela u odnosu na CNN model, naročito pri višim SNR vrednostima, gde njegova sposobnost izdvajanja robusnih karakteristika dovodi do superiornih performansi.

Međutim, s obzirom da oba modela pokazuju slične slabosti pri razlikovanju modulacija sa kvadraturnim amplitudskim osobinama i WBFM, to ukazuje na potrebu za daljom optimizacijom, predobradom signala ili primenom složenijih modela kako bi se dodatno unapredila tačnost klasifikacije.



Slika 6. *Prosečna tačnost klasifikacije za dva ispitivana modela.*

6. LITERATURA

[1] Y. Wang, M. Liu, J. Yang, G. Gui, "Data-Driven Deep Learning for Automatic Modulation Recognition in Cognitive Radios", IEEE Transactions on Vehicular Technology, vol. 68, no. 4, pp. 4074-4077, April 2019. Dostupno na: https://ieeexplore.ieee.org/document/8645696.

[2] Z. Ke, H. Vikalo, "Real-Time Radio Technology and Modulation Classification via an LSTM Auto-Encoder", Novembar 2020, Dostupno na: https://arxiv.org/abs/2011.08295.

[3] D. P. Kingma, J. Ba, "Adam: A Method for Stochastic Optimization", Januar 2017. Dostupno na: https://arxiv.org/abs/1412.6980

[4] J. L. Ziegler, R. T. Arn, W. Chambers, "Modulation recognition with GNU radio, keras, and HackRF", Mart 2017. Dostupno na: https://ieeexplore.ieee.org/document/7920747.

Kratka biografija:

Milica Jankov rođena je u Zrenjaninu 2000. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Telekomunikacioni sistemi odbranila je 2024.god.

Kontakt: milicajankov@uns.ac.rs

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.4
DOI: https://doi.org/10.24867/31BE15Topalovic

PREDIKCIJA CENA AVIONSKIH LETOVA UPOTREBOM ALGORITAMA MAŠINSKOG UČENJA

AIRLINE FLIGHT PRICE PREDICTION USING MACHINE LARNING ALGORITHMS

Saška Topalović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIČKO I RAČUNARSKO INŽENJERSTVO

Kratak sadržaj – *U ovom radu se istražuje problem predikcije cena letova. Predikcija cena avionskih letova predstavlja izazov kako za putnike, koji žele da plaćaju objektivnu cenu, tako i za avio-kompanije, koje nastoje da optimizuju prihode. Tradicionalne metode predikcije često ne uzimaju u obzir varijabilne faktore poput sezonskih oscilacija, potražnje i dostupnosti sedišta, što rezultuje neadekvatnim cenama. Stoga je razvijanje modela zasnovanog na mašinskom učenju važno kako bi se postigla veća transparentnost i efikasnost tržišta avionskih karata. Metodologija rada uključuje upotrebu sledećih algoritama mašinskog učenja: Random Forest, Decision Tree i XGBoost. Za evaluaciju modela su korišćeni trening i test skup podataka u odnosu 8:2. Evaluacijom performansi, pomoću RMSE (Root Mean Squared Error) metrike, Random Forest se pokazao kao najbolji model za predikciju cena avionskih karata.*

Ključne reči: *cena avionskog leta; mašinsko učenje; Random Forest; Decision Tree; XGBoost;*

Abstract – *This paper investigates the problem of flight price prediction. The prediction of flight prices poses a challenge for both passengers, who seek to pay a fair price, and airlines, which aim to optimize their revenues. Traditional prediction methods often fail to account for variable factors such as seasonal fluctuations, demand, and seat availability, resulting in inaccurate pricing. Therefore, developing a machine learning-based model is essential for achieving greater transparency and efficiency in the flight ticket market. The methodology employed includes the use of the following machine learning algorithms: Random Forest, Decision Tree, and XGBoost. The model evaluation was performed using an 8:2 train-test data split. Based on the performance evaluation using the RMSE (Root Mean Squared Error) metric, Random Forest proved to be the best model for flight price prediction..*

Keywords: *airline flight price; machine learning; Random Forest; Decision Tree; XGBoost;*

1. UVOD

Tržište avionskih karata je veoma dinamično i podložno brojnim promenama, što direktno utiče na cene avionskih letova, a samim tim i na putnike, avio-kompanije i druge relevantne aktere u industriji. Putnici nastoje da pronađu najpovoljnije avionske karte i tako optimizuju svoj budžet, a predikcija cena avionskih letova bi im omogućila da unapred planiraju i rezervišu svoja putovanja sa boljim razumevanjem očekivanih cena. Avio-kompanije bi, s druge strane, mogle da koriste predikciju cena letova kako bi bolje upravljale svojim tarifama, prilagodile ih potrebama potražnje i konkurencije, te optimizovale svoje prihode.

Razvoj modela za predikciju cena avionskih letova predstavlja veliki izazov zbog širokog spektra tehničkih i tržišnih faktora. Fokus ovog rada je implementacija algoritma mašinskog učenja koji će moći efikasno da analizira i obrađuje različite podatke kao što su karakteristike letova, istorijske cene i trendovi potražnje. Cilj je postići što veću preciznost u predikciji cena letova, uzimajući u obzir složenu i dinamičnu prirodu ovog tržišta.

Za kreiranje modela korišćena su tri algoritma mašinskog učenja: *Random Forest, Decision Tree* i *XGBoost*. Modeli su obučeni na podacima koji sadrže različite tehničke i tržišne karakteristike letova, kao što su vreme polaska, destinacija, avio-kompanija, broj presedanja, trajanje leta i istorijske cene. Evaluacija modela izvršena je podelom podataka na trening i test skupove u odnosu 8:2. Za merenje uspešnosti modela korišćena je RMSE (*Root Mean Squared Error*) metrika. Na osnovu dobijenih rezultata, *Random Forest* algoritam se istakao kao najprecizniji model za predviđanje cena letova, nadmašivši ostale modele po tačnosti.

U nastavku rada biće detaljno objašnjeni različiti aspekti rešavanog problema. U poglavlju 2 daje se pregled radova koji se bave sličnom tematikom. Poglavlje 3 sadrži opis skupa podataka, eksplorativnu analizu i pretprocesiranje podataka, kao i algoritme korišćene za predikciju cena letova. Poglavlje 4 je posvećeno diskusiji rezultata dobijenih primenom formiranih modela. Na kraju, poglavlje 5 sadrži zaključak koji rezimira ključne uvide.

2. SRODNA ISTRAŽIVANJA

U radu [1], Tziridis i saradnici su identifikovali skup karakteristika koje opisuju tipičan let, pretpostavljajući da one utiču na cenu. Studija se sastoji iz 4 faze. U prvoj fazi su eksperimentalno određena obeležja koja utiču na cenu primenom "*oneleave-out*" pravila. U drugoj fazi ručno su prikupljeni podaci sa veba, ukupno 1814 letova, gde je svaki let opisan sa 8 obeležja. U trećoj fazi odabrano je 8 modela mašinskog učenja i primenjeno na iste podatke. Na kraju, u četvrtoj fazi rešenje je evaluirano desetostrukom

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

unakrsnom validacijom. Mera performansi korišćena za upoređivanje modela je MSE. Eksperimenti su pokazali da je uklanjanjem karakteristike „broj preostalih dana do poletanja“ rezultat bio najgori. Pored toga, *Bagging Regressor* i *Random Forest* su uvek imali dobre performanse. Rezultati su pokazali da su modeli mašinskog učenja zadovoljavajući za predviđanje cena avionskih letova i da dostižu tačnost čak do 88%.

Wang i saradnici predložili su novi okvir mašinskog učenja za predviđanje kvartalne prosečne cene avionskih karata na nivou tržišnog segmenta, fokusirajući se na određene kombinacije polazišta i odredišta [2]. Koristili su kombinaciju dva javno dostupna skupa podataka: *Airline Origin and Destination Survey* (DB1B) i *Air Carrier Statistics database* (T-100). Podaci su prikupljeni tokom 2018. godine. DB1B skup podataka obuhvata informacije o avio-liniji, dok T-100 skup podataka sadrži statističke podatke o avio-prevoznicima. Za evaluaciju modela koristili su RMSE i prilagođeni R kvadrat. Za kreiranje modela mašinskog učenja koristili su *Random Forest*, linearnu regresiju, SVM, *Multilayer Perceptrons* (MLPs) i *XGBoost Tree*. Zahvaljujući tehnikama selekcije obeležja, pokazalo se da je *Random Forest* model sposoban da predviđa kvartalnu prosečnu cenu avio-karte sa prilagođenim R kvadratom od 0.869.

3. METODOLOGIJA

U ovom poglavlju će biti predstavljena implementacija sistema za predikciju cena avionskih letova. Ulaz u sistem čini skup podataka o letovima, uključujući informacije o datumu polaska, avio-kompaniji, vremenu polaska i dolaska, broju presedanja i drugim faktorima. Izlaz sistema su predviđene cene karata na osnovu modela mašinskog učenja, koji su obučeni na prethodno pripremljenim podacima.

Poglavlje je organizovano u nekoliko potpoglavlja. Prvo će, u potpoglavlju 3.1, biti opisan skup podataka, uključujući njegove osnovne karakteristike. Zatim će, u potpoglavlju 3.2 biti opisana ekplorativna analiza i pretprocesiranje podataka. Dok će se poglavlje 3.3 fokusirati na obuku modela i optimizaciju hiperparametara.

3.1. Skup podataka

Skup podataka korišćen u radu je “*Flight Fare Prediction*”, javno dostupan, preuzet sa *Kaggle* platforme [3]. Sadrži informacije o avionskim letovima iz 2019. godine i sastoji se od ukupno 10683 zapisa. Svaki let u skupu podataka opisan je sa jedanaest obeležja, a to su: avio-kompanija (Airline), datum putovanja (Date_of_Journey), polazište (Source), odredište (Destination), ruta (Route), vreme polaska (Dep_Time), vreme dolaska leta (Arrival_Time), trajanje leta (Duration), ukupan broj presedanja tokom putovanja (Total_Stops), dodatne informacije o putovanju (Additional_Info), cena avionske karte (Price) Cena avionske karte predstavlja ciljno obeležje, tj. zavisnu promenljivu koja se predviđa na osnovu preostalih deset nezavisnih obeležja.

3.2. Eksplorativna analiza i pretprocesiranje podataka

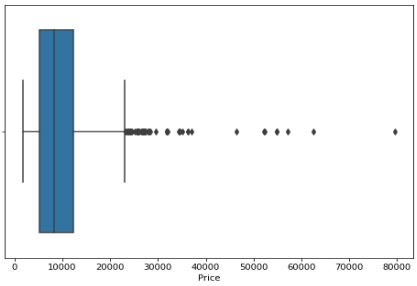
Najpre je izvršena analiza nedostajućih vrednosti, pri čemu je ustanovljeno prisustvo nedostajućih obeležja (NaN) u skupu podataka. Pošto je samo jedan let imao dva nedostajuća obeležja, on je uklonjen iz skupa podataka. Nakon toga, izvršena je identifikacija duplikata i ustanovljeno je da postoji 220 potpuno identičnih zapisa. Prisustvo više identičnih instanci ne doprinosi raznolikosti i kvalitetu podataka, već samo povećava nepotrebnu složenost. Da bi se rizik od grešaka i nepreciznosti sveo na minimum, ti duplikati su uklonjeni.

Zatim, neophodna je bila konverzija vremenskih obeležja. Promenljive *Date_of_Journey*, *Dep_Time*, *Arrival_Time* i *Duration* konvertovane su iz tipa *string* u tip *datetime*. Nakon ove konverzije, izvedena su nova obeležja dan (engl. *Day_of_Journey*) i mesec putovanja (engl. *Month_of_Journey*), sati (engl. *Dep_Hours*) i minuti (engl. *Dep_Minutes*) polaska, kao i vreme dolaska (engl. *Arrival_Hours* i *Arrival_Minutes*). Pored toga, izračunato je i ukupno vreme trajanja leta u minutama (engl. *Duration_Minutes*), što je omogućilo bolje razumevanje i analizu vremenskih obrazaca u podacima, čineći ih pogodnijim za dalje modelovanje.

Cena avionskih letova može varirati u zavisnosti od dana u sedmici, vremena polaska i meseca putovanja. Letovi vikendom i tokom turističke sezone ili praznika često imaju više cene, dok letovi kasno uveče ili noću, zbog manje potražnje, mogu biti jeftiniji. Analiza prosečnih cena pokazala je najpre da su cene letova vikendom neznatno više u odnosu na cene radnim danima, kao i da se nijedan dan u sedmici posebno ne izdvaja, ali su petkom i nedeljom cene blago više u odnosu na ostale dane. Pretpostavka da su noćni letovi (između 19h i 7h) u proseku nešto jeftiniji od dnevnih se pokazala tačnom. Najviša prosečna cena bila je u martu, a najniža u aprilu, prema podacima iz skupa koji obuhvataju period od marta do juna.

Uobičajena pojava u avio-industriji je da veći broj presedanja tokom putovanja obično rezultuje višim cenama avionskih karata, a analiza je to i potvrdila. Letovi sa četiri presedanja su značajno skuplji od direktnih letova.

Kako je cena jedino numeričko obeležje u skupu podataka, bilo je važno ispitati moguća odstupanja, odnosno *outlier*-e. U svrhu identifikacije *outlier*-a je korišćen *boxplot* grafikon, koji je pokazao da postoji šest cena koje se značajno razlikuju od ostalih (Slika 1).



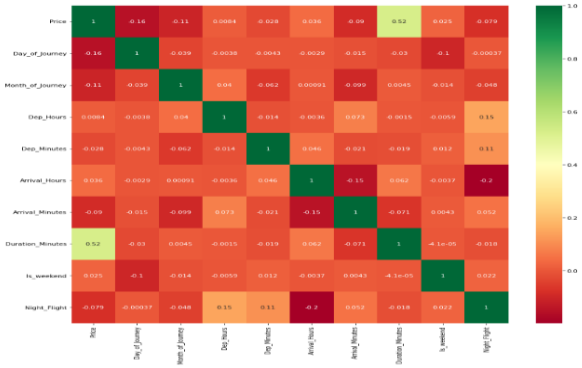
Slika 1. *Raspodela cena u skupu podataka*

Uočavanjem ovih *outlier*-a, sledeći korak je bio analiza prosečne cene letova po avio-kompanijama kako bi se utvrdilo da li neki od tih letova pripada *Business* klasi, što bi moglo objasniti njihovo odstupanje. Ispostavilo se da postoje letovi *Business* klase avio-kompanije *Jet Airways*, koji znatno odstupaju po ceni. Kako je u pitanju samo šest takvih letova, zaključeno je da, zbog malog broja primeraka, nije svrsishodno vršiti posebnu predikciju cena

za ovu klasu letova, pa su ti letovi uklonjeni iz skupa podataka.

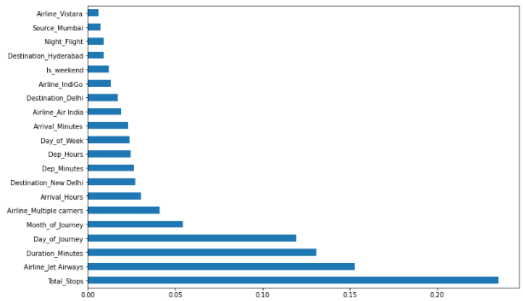
U cilju boljeg razumevanja raspodele kategoričkih obeležja, generisani su *barplot* dijagrami za svaku kategoriju. Posebno se izdvojio dijagram za promenljivu '*Additional_Info*', gde je oko 80% podataka pripadalo kategoriji '*No Info*'. Takođe, promenljiva '*Route*' pokazala je značajnu povezanost sa brojem presedanja, polazištem i odredištem putovanja, što smanjuje njenu nezavisnost. Kako ove dve promenljive ne doprinose informativnosti i raznolikosti, one su izuzete iz dalje analize.

Za analizu korelacije korišćen je *heatmap* kao vizuelni alat. Najviša korelacija, od 0.52, uočena je između cene avionskih letova i trajanja putovanja, što ukazuje na umerenu pozitivnu vezu između ove dve promenljive (Slika 2). To znači da duži letovi, u proseku, imaju više cene u odnosu na kraće letove. Ipak, važno je napomenuti da korelacija samo ukazuje na to da postoji neka vrsta povezanosti, ona ne implicira uzročno-posledičnu vezu između faktora. Ostala obeležja su pokazala značajno niže korelacije.



Slika 2. *Matrica korelacije skupa obeležja*

Za identifikaciju najznačajnijih obeležja u predviđanju cene letova korišćen je *Extra Tree Regressor*. Na osnovu rezultata prikazanih na Slika 3, utvrđeno je da je '*Total_Stops*', odnosno broj presedanja, najznačajnije obeležje sa najvećim uticajem na cenu leta.



Slika 3. *Vizualizacija 20 najvažnijih obeležja*

Važan korak pretprocesiranja je enkodovanje kategoričkih obeležja, budući da mnogi algoritmi mašinskog učenja zahtevaju da su ulazni podaci u numeričkom obliku. Za enkodovanje nominalnih kategoričkih obeležja (*Airline*, *Source* i *Destination*) korišćen je *one-hot encoding*. Broj presedanja (engl. *Total_Stops*) je jedino ordinalno kategoričko obeležje i za njegovo enkodovanje korišćen je *label encoding*, pri čemu su redni brojevi dodeljeni svakoj kategoriji, od '*non-stop*' (0) do najvećeg broja presedanja.

Na kraju, skup podataka je sortiran po datumu putovanja. Ovakvo sortiranje omogućilo je da se test skup formira kao vremenski interval nakon trening skupa, čime je simulirano predviđanje budućih cena na osnovu istorijskih podataka, što dodatno povećava verodostojnost modela.

3.3. Obuka modela i optimizacija hiperparametara

Za potrebe obuke modela, odabrana su tri algortima mašinskog učenja *Random Forest*, *Decision Tree* i *XGBoost*. *Decision Tree* algoritam pravi predikcije kroz seriju binarnih odluka, postepeno deleći skup podataka na podskupove na osnovu karakteristika koje najviše doprinose rešenju problema. *Random Forest* obučava stotine stabala odluke na različitim uzorcima podataka, a konačni izlaz je prosek vrednosti svih modela ili odluka dobijena većinskim glasanjem, u zavisnosti od tipa problema (regresija ili klasifikacija). *XGBoost* primenjuje tehniku *boosting*-a, tj. integriše veliki broj slabih klasifikatora kako bi formirao snažan model.

Svaki od pomenutih algoritama obučen je na pretprocesiranim podacima korišćenjem *test-train* podele u odnosu 8:2. Mera korišćena za evaluaciju performansi modela je RMSE (*Root Mean Squared Error*).

Nakon inicijalne obuke, izvršena je optimizacija hiperparametara. U tu svrhu korišćena je tehnika *RandomizedSearchCV*, koja omogućava ispitivanje različitih kombinacija hiperparametara unutar unapred definisanih opsega. Ovaj pristup ispituje nasumično izabrane kombinacije, čime se značajno ubrzava proces optimizacije u poređenju sa klasičnim *GridSeachCV*-om. Za svaki model, proces optimizacije hiperparametara obuhvatao je 10 iteracija slučajnog pretraživanja. Za svaku kombinaciju hiperparametara primenjen je kros-validacioni proces sa 5 podela (engl. *5-fold cross-validation*), gde je glavna metrika za ocenjivanje modela bila RMSE.

4. REZULTATI I DISKUSIJA

Prethodno obučeni modeli su testirani na skupu od 2092 leta koja nisu bila uključena u trening skup. Rezultati, prikazani u TABELA I, pokazuju performanse sva tri modela pre i posle optimizacije hiperparametara, merene vrednostima RMSE.

TABELA I. PERFORMANSE MODELA PRE I POSLE OPTIMIZACIJE HIPERPARAMETARA

Model	**Pre optimizacije hiperparametara**	**Posle optimizacijom hiperparametara**
	RMSE	***RMSE***
Random Forest	2042.709	1899.209
Decision Tree	3345.024	2505.235
XGBoost	1908.776	1971.674

Pre optimizacije, *XGBoost* model je postigao najnižu RMSE vrednost od 1908.776, što ga čini najboljim modelom u početnim uslovima. Nakon optimizacije hiperparametara, *Random Forest* model je zabeležio poboljšanje, smanjivši RMSE na 1899.209, što ga čini najboljim modelom nakon optimizacije. Ovaj rezultat nije iznenađujući, s obzirom na to da se u radu [2] takođe pokazao kao model sa najboljim performansama,

nadmašivši *XGBoost*. Slično tome, u radu [1] su eksperimenti pokazali da *Random Forest* konzistentno pokazuje dobre rezultate u predikciji cena letova, što ukazuje na njegovu pouzdanost i stabilnost u različitim kontekstima. S druge strane, *XGBoost* model, koji je prvobitno imao najbolje rezultate, pokazao je blagi porast RMSE, ali je i dalje zadržao visoke performanse, dok se *Decision Tree* pokazao kao najlošiji model za rešavani problem.

Blagi pad rezultata nakon optimizacije hiperparametara, za *XGBoost* model je bio iznenađujući, međutim može se objasniti složenošću prostora hiperparametara koji se koriste, kao i brojem parametara koji su razmatrani tokom pretrage. *RandomizedSearchCV* algoritam je, zbog ograničenog broja iteracija, možda propustio ključne kombinacije koje bi dale bolje rezultate. Pored toga, nasumičnost algoritma može dovesti do neravnomernog istraživanja prostora parametara, što potencijalno znači da određene oblasti nisu bile adekvatno istražene. Ovo ukazuje na potrebu za detaljnijom pretragom prostora hiperparametara, kako bi se iskoristio pun potencijal *XGBoost* modela.

Analizom grešaka uočeno je da su sva tri modela ostvarila najveće greške u predikciji letova sa visokim cenama. Detaljnijom analizom otkriveno je da modeli imaju tendenciju da prave veće greške prilikom predviđanja cena za letove sa visokim cenama, ali relativno kratkim trajanjem. Ovaj fenomen može biti posledica uočene umerene pozitivne korelacije između cene avio-karte i trajanja leta, koja je detaljnije obrađena u poglavlju 3.2. Korelacija ukazuje na to da duži letovi obično imaju više cene, dok su kraći letovi često jeftiniji. Međutim, ove anomalije sugerišu da postoje specifični letovi gde visoke cene nisu u skladu sa očekivanjima na osnovu trajanja leta, što rezultuje većim greškama modela.

5. ZAKLJUČAK

U ovom radu predstavljen je sistem za predikciju cena avionskih letova. Putnici se često oslanjaju na procene cena letova koje pružaju turističke agencije ili veb portali za rezervaciju, ali te procene ne odražavaju uvek realne tržišne trendove. Takođe, zanemaruje se činjenica da avio-kompanije često prilagođavaju tarife različitim tržišnim faktorima. Razvoj modela za objektivnu procenu cena letova omogućio bi putnicima da plaćaju fer cenu, dok bi avio-kompanije mogle bolje prilagoditi svoje tarife tržišnim uslovima, čime bi poboljšale svoje poslovne performanse.

Primenom tehnika mašinskog učenja, razvijeni su modeli za predikciju cena letova korišćenjem algoritama *Random Forest*, *Decision Tree* i *XGBoost*. Nakon optimizacije hiperparametara, performanse modela su evaluirane metrikom RMSE, koja je izabrana zbog svoje osetljivosti na velike greške. Najbolje rezultate postigao je model *Random Forest*, sa RMSE vrednošću od 1899.209 na test podacima. Međutim, analiza grešaka pokazala je da modeli uglavnom greše pri predviđanju letova sa visokim cenama i kraćim trajanjem.

Ovaj rad doprinosi boljem razumevanju faktora koji utiču na cenu letova i postavlja osnovu za dalja istraživanja u oblasti predikcije cena avionskih letova. Dalji pravci istraživanja i potencijalna poboljšanja sistema mogli bi obuhvatiti primenu naprednijih tehnika optimizacije hiperparametara, kao što su *Bayesian Optimization* ili *GridSearchCV* sa većim brojem iteracija, kako bi se istražile sve ključne kombinacije. Takođe, predikcija cena je trenutno ograničena na korišćeni skup podataka, te bi razmatranje proširivanja skupa dodatnim relevantnim faktorima, poput broja dana do polaska, klase leta, pozicije sedišta u avionu ili broja besplatnog prtljaga, moglo doprineti tačnosti modela. Osim toga, analiza grešaka ukazuje na potrebu za prilagođavanjem modela u smislu boljeg tretiranja letova sa kraćim trajanjem i višim cenama, možda kroz uvođenje dodatnih relevantnih karakteristika ili kroz unapređenje algoritama koji su osetljiviji na ove specifičnosti.

6. LITERATURA

[1] Tziridis, K., Kalampokas, T., Papakostas, G. A., & Diamantaras, K. I. (2017, August). Airfare prices prediction using machine learning techniques. In *2017 25th European Signal Processing Conference (EUSIPCO)* (pp. 1036-1039). IEEE.

[2] Wang, T., Pouyanfar, S., Tian, H., Tao, Y., Alonso, M., Luis, S., & Chen, S. C. (2019, July). A framework for airfare price prediction: a machine learning approach. In *2019 IEEE 20th international conference on information reuse and integration for data science (IRI)* (pp. 200-207). IEEE.

[3] https://www.kaggle.com/datasets/nikhilmittal/flight-fare-prediction-mh (pristupljeno u martu 2024.)

Kratka biografija:

Saška Topalović je rođena 31.08.2000. godine u Doboju, gde je stekla svoje osnovno i srednje obrazovanje. Školske 2019/20. godine se upisuje na Fakultet tehničkih nauka u Novom Sadu na studijski program softversko inženjerstvo i informacione tehnologije. Diplomski rad pod nazivom „Arhitektura nultog poverenja“ odbranila je 2023. Iste godine se upisuje na master akademske studije, na isti studijski program. 2024. je položila sve ispite predviđene planom i programom i stekla uslov za odbranu master rada.

kontakt: tsaska98@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.38
DOI: https://doi.org/10.24867/31BE16Halgato

PROJEKTOVANJE IoT UREĐAJA SA MOGUĆNOŠĆU PRIKUPLJANJA ENERGIJE SA PROVODNIKA INDUSTRIJSKIH INTERFEJSA

DESIGN OF IoT DEVICE WITH THE ABILITY TO HARVEST ENERGY FROM THE LINES OF AN INDUSTRIAL INTERFACE

Andraš Halgato, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je opisan IoT uređaj sa mogućnošću prikupljanja energije sa provodnika industrijskih interfejsa. Predstavljene su različite metode za ograničavanje struje. Detaljno je prikazano projektovanje kola za strujno ograničavanje, koje je nezavisno od veličine ulaznog napona. Takođe, opisane su procedure za merenje vremena potrebnog za punjenje superkondenzatora. Prikazani su simulacioni rezultati napona i struja od interesa pri različitim uslovima, i merni rezultati pri korišćenju uređaja u realnom okruženju.*

Ključne reči: *IoT, superkondenzator, simulacije, štampana kola, strujno ograničavanje*

Abstract – *This paper describes a smart device capable of harvesting energy from the lines of industrial interfaces. The implementation of a current limiting circuit, which is independent of the intensity of the input voltage, is explained in detail. Various methods for current limiting are presented. Several procedures for measuring the time required to charge a supercapacitor are described. Simulation results of voltages and currents of interest under various conditions, and measurement results when using the device in a real environment, are shown.*

Keywords: *IoT device, supercapacitor, simulation, printed circuit board, current limiting*

1. UVOD

Električni sistemi sa malom potrošnjom energije, projektovani na bazi silicijuma, stvorili su neke potpuno nove mogućnosti u razvoju elektronike. Obezbedili su razvoj minijaturnih, prenosivih, pa čak i modularnih elektronskih uređaja. Broj umreženih uređaja (eng. *Internet of Things*) se veoma brzo povećava u svakodnevnom životu. Sve više elektronskih uređaja se povezuje na internet mrežu. Svi ovi uređaji očitavaju ili šalju neke podatke preko mreže i koriste skladištenje podataka na oblaku (eng. *Cloud Storage*).

Ovakvi uređaji uglavnom imaju malu potrošnju električne energije. Ne zahtevaju velika napajanja, ali ipak, energija se mora nekako dovesti do njih i, pri projektovanju sistema, najveći izazov predstavlja realizacija napajanja ovih električnih kola.

Svi ovi uređaji zahtevaju neku vrstu kompaktnog, jeftinog i lakog izvora napajanja, da bi se osigurala prenosivost i autonomija. Prvenstveno se koriste baterije za akumulisanje energije, čiji se kapacitet drastično povećao u prethodnih 15 godina. Pojavile su se i mnoge druge alternative za skladištenje energije, međutim nijedna druga metoda ne može da obezbedi toliki kapacitet za skladištenje električne energije kao baterije ili akumulatori. Alternativno rešenje za skladištenje naelektrisanja predstavlja superkondenzator koji se sve češće primenjuje, kako kod malih umreženih uređaja, tako i kod električnih automobila, a i u većim – zahtevnijim električnim sklopovima [1].

Uređaji za prikupljanje energije koji pretvaraju ambijentalnu energiju u električnu privukli su veliko interesovanje u vojnom i komercijalnom sektoru. Energija se takođe može sakupljati za napajanje malih autonomnih senzora, kao što su senzori razvijeni u MEMS tehnologiji. Ovi sistemi su često vrlo mali i zahtevaju malo energije, ali njihove primene su ograničene oslanjanjem na baterijsku energiju. Sakupljanje energije iz ambijentalnih vibracija, vetra, toplote ili svetlosti moglo bi omogućiti pametnim senzorima da funkcionišu neograničeno [2].

U ovom radu je prikazan IoT uređaj sa mogućnošću prikupljanja energije sa provodnika industrijskih interfejsa.

2. PROJEKTOVANJE KOLA ZA OGRANIČAVANJE STRUJE

Cilj rada je razvijanje IoT uređaja koji prikuplja energiju sa vodova (provodnika) industrijskih interfejsa, koristeći superkondenzator za skladištenje energije. Uređaj treba da se napaja iz postojećih industrijskih mreža, konkretno sa Meter Bus linija za prenos podataka, omogućavajući mu autonomno funkcionisanje bez eksternog izvora napajanja.

Ključan deo uređaja je električno kolo koje ograničava potrošnju na maksimalno 15 mA, bez obzira na varijacije napona napajanja. Sve komponente, uključujući ESP8685 Espressif mikrokontroler [3], pažljivo su odabrane kako bi minimizovale potrošnju energije, čineći uređaj energetski efikasnim za industrijska okruženja. Mikrokontroler omogućava daljinsko praćenje i upravljanje putem

NAPOMENA: Ovaj rad proistekao je iz master rada čiji mentor je bila dr Mirjana Damnjanović, red. prof.

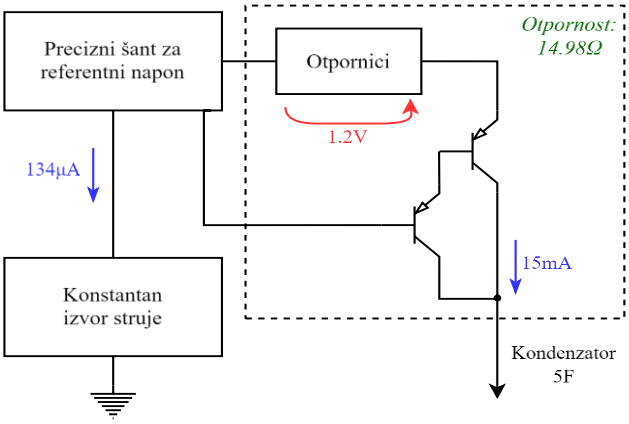
interneta. Ova funkcionalnost je važna jer omogućava daljinsko praćenje mernih podataka i upravljanje uređajem putem internet mreže. Takođe, uređaj obezbeđuje očitavanje određenih parametara sa Meter Bus linija i bežično prenošenje podataka ka centralizovanom sistemu za analizu.

Postoje mnogobrojna rešenja za ograničavanje strujne potrošnje nekog uređaja [4], [5]. Ipak, nijedno od tih rešenja nema mogućnost ograničenja strujne potrošnje, tako da bude nezavisna od veličine ulaznog napona. Nakon sprovođenja brojnih eksperimenata, s ciljem ograničenja strujne potrošnje na 15 mA, izabrane su integrisane komponente LM334 i LM4041 sa dodatnim tranzistorima i diodama za postizanje odgovarajuće strujne potrošnje (slika 1).

Podešavanjem vrednosti struje koju LM334 odvodi sa priključka povratne sprege (*eng. Feedback Pin*) LM4041 komponente i formiranjem napona otpornicima u grani "*feedback*" priključka, dobijaju se dve konstantne vrednosti (konstantno odvođenje struje i konstantan napon na otpornicima). Ove vrednosti se ne menjaju promenom veličine ulaznog napona, pa samim tim neće se menjati ni strujna potrošnja. Dakle, upotrebom ovih komponenti je u istom trenutku pokriven širok opseg ulaznih napona, ali je i strujna potrošnja ograničena na 15 mA.

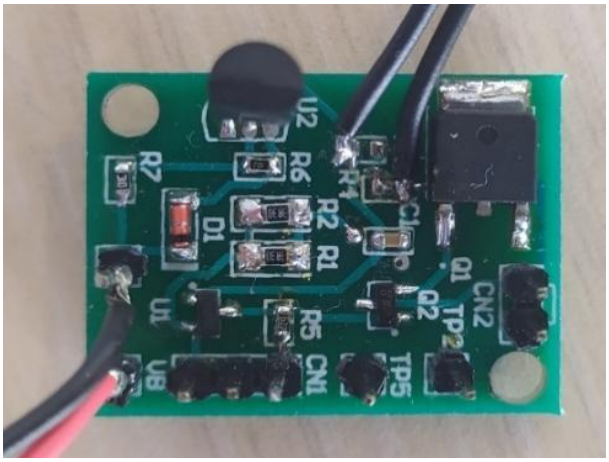
Obezbeđivanjem širokog opsega ulaznog napona, postiže se veća fleksibilnost i univerzalnost uređaja, jer se na taj način otvaraju neke druge mogućnosti punjenja superkondenzatora. Strujno ograničenje se uvodi zato što se uređaj napaja samo superkondenzatorom od 5 F, tj. napaja se bez vlastitog izvora energije, kao što je baterija ili mrežni adapter. Sve dok se kondenzator ne napuni do neke granične vrednosti, ESP8685 ostaje bez nominalnog napajanja, te neće ništa očitavati niti slati.

Kolo za ograničavanje struje se sastoji od dva dela. U prvom delu je projektovano napajanje uređaja. Ovo kolo obezbeđuje napajanje ESP8685 modulu, pomoću kojeg se šalju odgovarajući podaci preko internet mreže. Međutim, pored ovih funkcija, najbitnija funkcionalnost ovog dela jeste podešavanje konstantnog napona i potrošnje struje na određene vrednosti. Fotografija dela realizovanog IoT uređaja koji služi za ograničavanje struje je prikazana na slici 2.



Slika 1. - *Blok šema kola za ograničavanje struje korišćenjem LM4041 i LM334 integrisanih kola*

U drugom delu kola se nalazi sam ESP8685 mikrokontroler. Za uspostavljanje veze između glavnog i sporednog dela uređaja na M-Bus magistrali, korišćena je TSS721 komponenta, koji ima ulogu da prilagodi naponske nivoe glavnog i zavisnog uređaja. Drugim rečima, ovaj deo koristi se za podešavanje nivoa napona logičke "1" i "0". Dodatno, obezbeđuje galvansku izolaciju zavisnih uređaja sa optokaplerima (eng. *Optocoupler*).



Slika 2. *Štampana pločica kola za ograničavanje strujne potrošnje*

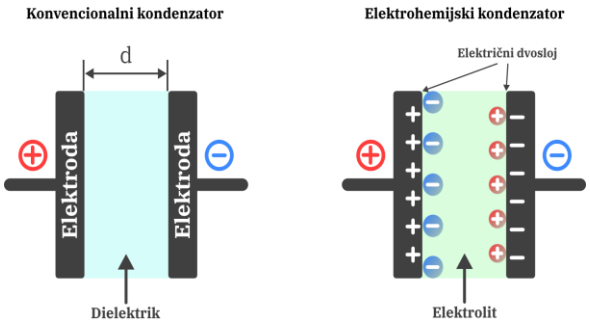
3. SKLADIŠTENJE ENERGIJE POMOĆU SUPERKONDENZATORA

Superkondenzatori, koji se nazivaju još i kondenzatorima sa visokim kapacitetom ili ultrakondenzatorima, smatraju se komponentama koje premošćuju jaz između elektrolitskih kondenzatora i punjivih baterija (slika 3), [1]. Prvobitno su ih razvijali u vojne svrhe, kasnije su se unapređivali za upotrebu u električnim automobilima. Mogu da akumulišu od 10 do 100 puta više energije od klasičnih elektrolitskih kondenzatora.

Očigledna prednost ove komponente je da se može napuniti brže nego punjiva baterija. Dodatno, mogu pretrpeti znatno veći broj cikličnih punjenja i pražnjenja. Mogu se podeliti na tri glavne grupe:

1. elektrohemijske kondenzatore (EDLC),
2. pseudokondenzatore (PC) i
3. hibridne kondenzatore.

Superkondenzatori se zasnivaju na dva elektrohemijska fenomena, na elektrohemijskom dvojnom sloju i elektrohemijskoj difuziji.



Slika 3. - *Ilustracija konvencionalnog i elektrohemijskog kondenzatora [1]*

4. PROJEKTOVANJE IoT UREĐAJA SA MOGUĆNOŠĆU PRIKUPLJANJA ENERGIJE

Savremena pametna digitalna brojila za električnu energiju, koji se često zovu i modularnim pametnim brojilima, obično raspolažu i nekim dodatnim priključcima za prenos podataka pomoću odgovarajućeg komunikacionog protokola. Neki od takvih prototola su M-Bus protokol [6] ili P1 serijski komunikacioni protokol.

Ovi priključci primarno služe za povezivanje uređaja za očitavanje stanja električnog brojila i korišćeni su za realizaciju predloženog IoT uređaja. Predviđen je i USB priključak za punjenje superkondenzatora. Pomoću njega uređaj može da se dovede u spremno stanje, gde će moći da radi obradu podataka čim se poveže sa nekim sistemom.

Princip rešenja opisanog u ovom radu se zasniva na korišćenju superkondenzatora za sakupljanje i skladištenje energije [2], [7].

Podrazumeva se da povećanje ili smanjenje kapacitivnosti superkondenzatora može uticati na vreme rada pametnog uređaja, što utiče i na period očitavanja i slanja podataka.

Da bi se izračunalo vreme punjenja kondenzatora, mora se uzeti u obzir i vremenska konstanta τ električnog kola. Uzimajući u obzir otpornost tranzistora, ukupna otpornost preko koje se puni kondenzator je:

$$R = 1{,}65\ \Omega + 13{,}33\ \Omega = 14{,}98\ \Omega, \qquad (1)$$

tako da je vremenska konstanta:

$$\tau = R \cdot C = 14{,}98\ \Omega \cdot 5\text{F} = 74{,}9\ \text{s}. \qquad (2)$$

U idealnom slučaju, smatra se da će kondenzator od 5 F biti potpuno napunjen nakon vremena smirivanja, tj. $5 \cdot \tau = 374{,}5$ s. Ovo važi samo u slučaju da nema strujnog ograničenja.

Ukoliko se uzme u obzir i strujno ograničenje od 15 mA, dobija se nešto drugačija vrednost za vreme punjenja.

Ako se pretpostavi da se kondenzator puni napajanjem od 5 V, i ako je u kolo, u paraleli, priključena i jedna cener dioda za zaštititu kondenzatora od prekomernog punjenja čiji je napon proboja (eng. *Breakdown Voltage*) 4,7 V, interval punjenja kondenzatora T se može odrediti kao:

$$T = C \cdot \frac{U_0}{I_0} \qquad (3)$$

gde je U_0 vrednost napona koju treba da dostigne kondenzator, a struja I_0 je strujno ograničenje. Kada se u jednačinu uvrste konkretne vrednosti dobija se:

$$T = C \cdot \frac{U_0}{I_0} = 5\ \text{F} \cdot \frac{4{,}8\ \text{V}}{15\ \text{mA}} = 1600\ \text{s} \qquad (4)$$

Dakle, superkondenzator kapacitivnosti 5 F, u kolu gde postoji strujno ograničenje od 15 mA i potrošač od 14.98 Ω, potpuno će se napuniti za 26 minuta i 40 sekundi.

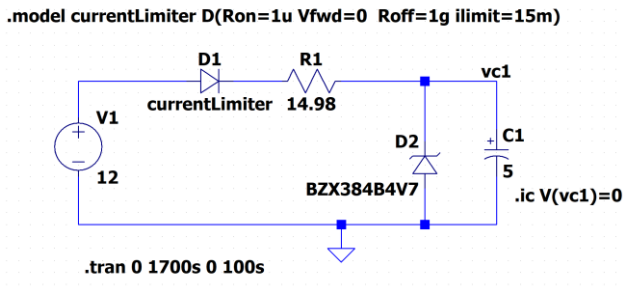
5. REZULTATI MERENJA I SIMULACIJA

Da li se izvršila simulacija rada projektovanog kola sa strujnim ograničenjem, korišćen je LTspice programski alat. Kolo koje se simulira je prikazano na slici 4, a rezultati simulacija na slici 5.

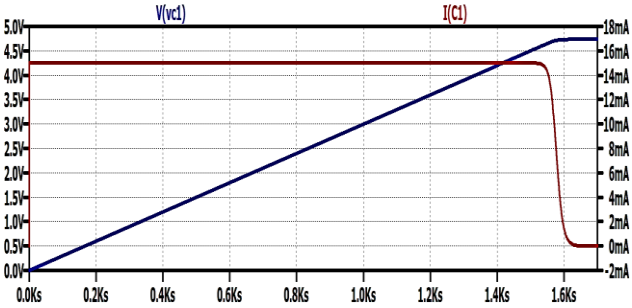
Treba napomenuti da je projektovano kolo za strujno ograničenje modelovano pomoću diode D1, koja u svom modelu ima ugrađen parametar za maksimalnu struju koju može da obezbedi.

Na osnovu simulacijom dobijenih vrednosti za napon na kondenzatoru (plava boja) i struju (crvena boja) može se zaključiti da se simulacije dobro slažu sa izračunatim vrednostima (jednačina (4)). Simulacijama određeno vreme punjenja je 1620 s.

Punjenje superkondenzatora prestaje kada se napuni na 4,7 V, odnosno kada se uključi Cener dioda koja sprečava dalje punjenje. Tada i struja punjenja kondenzatora naglo opada.



Slika 4. *Kolo za simulaciju punjenja superkondezatora C1*



Slika 5. *Simulacija punjenja superkondenzatora C1 naponom napajanja od 12 V (napon na kondenzatoru je prikazan plavom bojom, a struja crvenom)*

Nakon što su završeni svi neophodni proračuni i simulacije, IoT uređaj je realizovan, i izvršena su njegova testiranja. Testiranje je od ključnog značaja jer omogućava validaciju i verifikaciju rezultata dobijenih kroz proračune i simulacije.

Proces merenja je izveden korišćenjem prototipa "*smart meter*" uređaja, koji nije bio opremljen strujnim ograničavačem. Zato je posebno dizajniran strujni ograničavač, koji je korišćen isključivo za merenje i testiranje funkcionalnosti uređaja (slika 2). Merna postavka je bila sastavljena od jednog standardnog laboratorijskog napajanja i dva multimetra.

Uređaj je testiran pri naponima napajanja od 5 V, 8 V i 12 V. Napon napajanja od 5 V nije davao zadovoljavajuće rezultate, jer se kondenzator napunio samo do 3,74 V (slika 6). Kada je priključeno napajanje sa višim naponima, od 8 V i od 12 V, sam uređaj, kao i njegovo kolo za strujno ograničavanje, su radili na adekvatan način. Napon na kondenzatoru je dostigao 4,6 V (slika 7). Pri naponu napajanja od 12 V, izmerena je strujna potrošnja od 16, 94 mA, kao što je prikazano na slici 8.

Slika 6. *Izmerena vrednost napona na kondenzatoru pri napajanju od 5 V*

Slika 7. *Izmerena vrednost napona na kondenzatoru pri napajanju od 12 V*

Slika 8. *Izmerena strujna potrošnja IoT uređaja pri naponu napajanja od 12 V*

6. ZAKLJUČAK

U okviru ovog rada projektovano je strujno ograničavačko kolo IoT uređaja i analiziran je njegov rad pri različitim naponima napajanja. Ovo je važno jer IoT uređaj treba da ima strujno ograničenje, nezavisno od veličine napona koji se dovodi na njegov ulaz.

Testiranje uređaja je obuhvatalo i postupak merenja vremena punjenja superkondenzatora pri različitim naponima i strujnim granicama. Zaključeno je da napon od 5 V ne može da se koristi za napajanje uređaja, jer nije dovoljno visok da se kondenzator napuni.

Prilikom testiranja punjenja kondenzatora na 8 V i 12 V, strujna zaštita se aktivirala, a kondenzator se napunio do 4,6 V za 23 minuta i 7 sekundi, pri strujnom ograničenju od 15 mA.

Izvršena su merenja i sa strujnim ograničenjem od 25 mA i naponom napajanja od 12 V. U tom slučaju, kondenzator se brže napunio do 4,6 V – za 15 minuta i 2 sekunde.

U budućem radu, moguće je dalje unaprediti rad uređaja i istražiti sledeće mogućnosti:

1. dodatno smanjiti potrošnju uređaja, korišćenjem komponenti koje imaju još manju potrošnju (mikrokontroler, tranzistori i ostale integrisane komponente),
2. optimizacija programskog koda mikrokontrolera,
3. istražiti mogućnost punjenja superkondenzatora putem provodnika nekog drugog komunikacionog protokola,
4. smanjivanje dimenzija štampane ploče uređaja.

Ono što bi u budućem radu trebalo ispitati je i međusobni uticaj komponenti. Pri tome, treba voditi računa da li će se uvođenjem ovih modifikacija pojaviti i neki neželjeni efekti, čime bi se narušila efikasnost kola.

LITERATURA

[1] Internet stranica: „Supercapacitor”, https://en.wikipedia.org/wiki/Supercapacitor, pristupljeno: mart 2024.

[2] DigiKey’s European Editors: „Energy Harvesting for Industrial Networking”, 2012, Internet stranica: https://www.digikey.com/en/articles/energy-harvesting-for-industrial-networking, pristupljeno: mart 2024.

[3] Tehnička dokumentacija: „ESP8685 Series“, Espressif Systems, 2024, internet stranica: https://www.espressif.com/sites/default/files/documentation/esp8685_datasheet_en.pdf, pristupljeno: jun 2024.

[4] NextPCB: „Current Limiting Circuits: A Complete Guide“, Internet stranica: https://www.nextpcb.com/blog/current-limiting-circuit, pristupljeno: jun 2024.

[5] Internet stranica: „Current limiting”, https://en.wikipedia.org/wiki/Current_limiting, pristupljeno: jun 2024.

[6] OMS Group: „M-Bus - The Standard for Remote Reading of Smart Meters“, 2020. Internet stranica: https://m-bus.com/, pristupljeno: mart 2024.

[7] „Energy harvesting”, Internet stranica: https://en.wikipedia.org/wiki/Energy_harvesting, pristupljeno: maj 2024.

Kratka biografija:

Andraš Halgato rođen je u Vrbasu 1997. god. Master rad na Fakultetu tehničkih nauka iz oblasti Energetika, elektronika i telekomunikacije – Primenjena elektronika odbranio je 2024.god.

Kontakt:
andras.hallgato@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.42
DOI: https://doi.org/10.24867/31BE17Maksimovic

UPOREĐIVANJE ALATA ZA KREIRANJE I SLANJE CARDANO TRANSAKCIJA

COMPARISON OF TOOLS FOR CREATING AND SUBMITTING CARDANO TRANSACTIONS

Miloš Maksimović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIČKO I RAČUNARSKO INŽENJERSTVO

Kratak sadržaj – *Sa rastućim brojem korisnika u blockchain sistemima, javlja se potreba za efikasnim alatima koji omogućavaju brže i pouzdanije kreiranje i slanje transakcija. U okviru Cardano blockchain sistema dostupno je nekoliko takvih alata. Cilj ovog rada je analizirati i uporediti performanse alata CLI, Ogmios i gOuroboros, kroz testiranje njihove efikasnosti u različitim scenarijima.*

Ključne reči: *Blokchain, Cardano, CLI, Ogmios, gOuroboros*

Abstract – *With the growing number of users in blockchain systems, there is an increasing need for efficient tools that enable faster and more reliable transaction creation and submission. Several such tools are available within the Cardano blockchain system. The goal of this paper is to analyze and compare the performance of the CLI, Ogmios, and gOuroboros tools by testing their efficiency in various scenarios.*

Keywords: *Blokchain, Cardano, CLI, Ogmios, gOuroboros*

1. UVOD

Jedan od najvećih izazova sa kojima se blokchain tehnologija suočava, još od njenog nastanka, je problem skalabilnosti. Kako se broj korisnika povećava, raste i broj digitalnih transakcija koje je neophodno obraditi u okviru sistema. Kada se sistem preoptereti transakcijama, dolazi do značajnog usporavanja zbog ogromne količine računarske moći neophodne za obradu svih transakcija.

U ovom radu će biti testirani alati za kreiranje i slanje transakcija na Cardano blokchain sistem. Među testiranim alatima se nalaze dva veoma popularna alata: CLI, koji je razvijen i održavan od strane Cardano developer-a i Ogmios, najpopularniji projekat otvorenog koda kreiran za ovu namenu. Pored ova dva alata, biće testiran i jedan novi projekat otvorenog koda pod nazivom gOuroboros.

Cilj ovog rada je testiranje broja kreiranih uspešno poslatih transakcija na Cardano blokchain korišćenjem prethodno navedenih alata i njihovo upoređivanje prema performansama i stabilnosti.

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

2. TEHNOLOGIJE

Blokchain je decentralizovan, distribuiran i nepromenljiv ledger obezbeđen kriptografskim heš algoritmima. Blokchain ledger sadrži lančano povezane hronološke i šifrovane blokove sinhronizovanih podataka preko Peer-to-Peer (P2P) mreže [1]. Podaci su formirani u blokove, a svaki blok je povezan sa prethodnim blokom time što se u okviru njega skladišti njegov heš. Blokchain prvo skladišti podatke u neverifikovane blokove, zatim se ti blokovi verifikuju uz pomoć distribuiranog konsenzus algoritma kako bi se održala sigurnost, privatnost i transparentnost u čitavoj blokchain mreži [2].

Ledger se može posmatrati kao baza u kojoj se vodi evidencija o transakcijama, dok se heš funkcija odnosi na funkciju koja za dati ulaz proizvodi izlaz takav da ne postoji drugi ulaz koji će proizvesti identičan izlaz. Transakcija je glavna operacija svakog blokchain sistema. Ona omogućava međusobni prenos sredstava ili podataka među korisnicima sistema.

2.1. Cardano

Cardano projekat je započet 2015. godine, a zvanično je i pušten u javnost septembra 2017. godine [3]. Inicijalno opisan kao blokchain sistem treće generacije, zamišljen je da reši probleme svojih prethodnika: skalabilnost, interoperabilnost i održivost. U Cardano ekosistemu, Ada je glavna valuta, ali postoji i manja jedinica nazvana Lovelace. Jedan Ada token je jednak milionu Lovelace-a.

Ouroboros je inovativni konsenzus protokol koji koristi Proof-of-Stake (PoS) mehanizam za postizanje konsenzusa u Cardano blockchain-u. Ključna karakteristika ovog protokola je koncept slotova i epoha [4]. Vreme u Cardano mreži je podeljeno na epohe, a svaka epoha je dalje podeljena na manje vremenske jedinice koje se nazivaju slotovi. U svakom slotu, jedan validator, nazvan slot leader, je izabran da kreira i dodaje novi blok u blockchain. Proces izbora slot leader-a se zasniva na količini Ada tokena koju korisnici stavljaju u zalog, čime se obezbeđuje decentralizovanost i sigurnost mreže.

2.2. CLI

Command Line Interface (CLI) je moćan alat za komandnu liniju koji omogućava korisnicima da interaguju sa Cardano blokchain mrežom. Pruža širok spektar funkcionalnosti za upravljanje tokenima, kreiranje i slanje transakcija, upite o informacijama mreže i rad sa stake pool-ovima. Kao esencijalna komponenta Cardano ekosistema, cardano-cli prvenstveno koriste programeri,

operateri stake pool-ova i napredni korisnici kojima je potreban direktan pristup osnovnim funkcijama mreže.

Cardano-cli funkcioniše komunicirajući direktno sa lokalnim Cardano čvorom, koji mora biti pokrenut i potpuno sinhronizovan sa mrežom da bi alat pravilno funkcionisao. Ova direktna komunikacija omogućava bezbedne i efikasne interakcije sa blokchain-om, jer se sve operacije obavljaju lokalno pre nego što se emituju na mrežu.

2.3. Ogmios

Ogmios je specijalizovani lightweight softverski most koji služi kao posrednik između Cardano čvorova i decentralizovanih aplikacija [5]. On je lightweight, što znači da nema prevelike zahteve za pokretanje u vidu računarskih resursa ili potrebu za velikom količinom prostora na disku računara jer ne koristi bazu podataka. Ovaj moćan alat dizajniran je da pojednostavi interakciju sa Cardano blokchain-om, pružajući programerima efikasan i pristupačan način za pristup podacima i funkcionalnostima mreže uz pomoć JSON/RPC (JavaScript Object Notation/Remote Procedure Call) poziva.

Za komunikaciju sa Cardano blokchain sistemom neophodan mu je pokrenuti i potpuno sinhronizovani lokalni čvor, slično kao i kod CLI-ja.

2.4. gOuroboros

gOuroboros je projekat otvorenog koda koji je pokrenula kompanija Blinklabs koja se bavi razvojem softvera vezanim za Cardano ekosistem. Ova biblioteka je razvijena korišćenjem programskog jezika Golang i ima mogućnost direktne komunikacije sa Cardano čvorom korišćenjem Ouroboros protokola.

Kao i kod prethodno navedenih alata, ovoj biblioteci je neophodan lokalno pokrenuti i potpuno sinhronizovani čvor sa kojim će komunicirati.

2.4. Grafana k6

Grafana k6 je moćan alat otvorenog koda za testiranje opterećenja, dizajniran da pomogne programerima, test inženjerima i analitičarima performansi u proceni performansi i pouzdanosti njihovih aplikacija, API-ja i infrastrukture pod različitim uslovima opterećenja [6].

U svojoj suštini, k6 je izgrađen da simulira virtuelne korisnike (virtual user - VU) koji mogu generisati saobraćaj za testiranje sistema. Ovi VU mogu izvršavati unapred definisane scenarije, oponašajući ponašanje i interakcije stvarnih korisnika sa aplikacijama. Ono što k6 izdvaja od mnogih drugih alata za testiranje opterećenja je njegov pristup usmeren ka programerima. Test skripte u k6 se pišu u JavaScript-u, koristeći ECMAScript 2015/ES6 standard, što ga čini pristupačnim širokom spektru programera koji su već upoznati sa veb tehnologijama. Proširivost k6 je olakšana kroz njegov sistem dodataka, koji korisnicima omogućava da prošire njegovu funkcionalnost kako bi zadovoljili specifične potrebe.

3. PROGRAMSKO REŠENJE

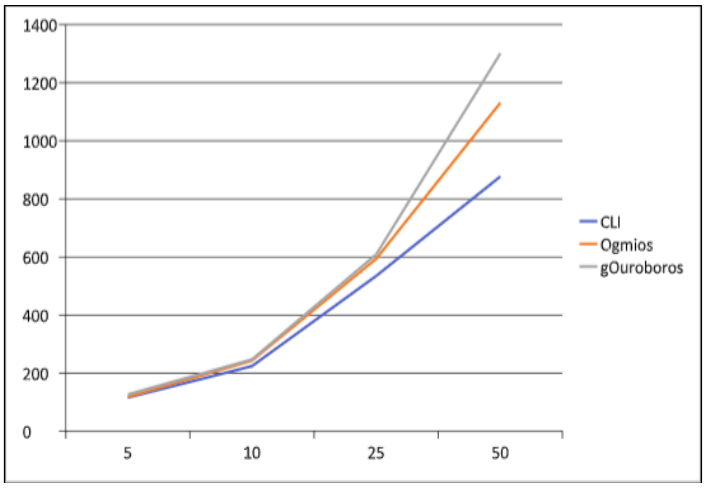
Za testiranje propusne moći CLI-ja, Ogmios-a i gOuroboros-a neophodno je kreirati veliki broj transakcija i poslati ih na Cardano blokchain pomoću ovih alata. Da bi se to postiglo korišćen je k6 alat za koji je kreirana nova ekstenzija i posebna skripta kojom je vršeno testiranje.

Svaki test je pokretan u vremenskom periodu od jednog minuta sa prethodno definisanim brojem VU i alatom koji će biti korišćen za kreiranje i slanje transakcija. S obzirom na to da svi alati zahtevaju lokalni sinhronizovani čvor, pre svakog testa se pokreće klaster od četiri čvora koji čine lokalni Cardano blokchain sistem.

Tokom svakog testa praćene su metrike poput broja uspešno poslatih transakcija za svakog VU, broj neuspešnih transakcija kao i koliko je ukupno transakcija uspešno poslato u toku trajanja testa.

4. REZULTATI

Svi testovi su pokretani sa prethodno opisanim scenarijom po tri puta za svaki od navedenih alata sa po 5, 10, 25 i 50 virtuelnih korisnika. Na slici 1. prikazan je dijagram uspešno kreiranih transakcija u toku trajanja testa po alatima. Ove vrednosti predstavljaju prosek dobijen nakon tri pokretanja.



Slika 1. *Ukupan broj uspešnih transakcija po alatu*

Iz datog dijagrama je moguće zaključiti da su pri manjim opterećenjima svi provajderi približno jednaki, ali kako opterećenje raste mogućnosti CLI-ja najviše opadaju. Ogmios i gOuroboros su poprilično izjednačeni sve dok ne dođe do velikih opterećenja gde gOuroboros pravi značajniju razliku i prednost u odnosu na Ogmios.

S obzirom na to da CLI već pri opterećenju od 50 virtuelnih korisnika prijavljuje značajniji broj neuspešnih transakcija njegovo dalje testiranje se čini suvišnim. Za razliku od njega Ogmios i gOuroboros ni pri 50 virtuelnih korisnika nisu trpeli dovoljno opterećenje koje bi dovelo do pojave neuspešnih transakcija zbog čega je imalo smisla napraviti još testova kojim će se proveriti njihova maksimalna propusnost. U tabeli 1 biće prikazani rezultati daljeg testiranja Ogmios-a i gOuroboros-a sa 75 virtuelnih korisnika.

Tabela 1. *Rezultati testiranja sa 75 virtuelnih korisnika*

	Ogmios	gOuroboros
Eksperiment 1	1323	1928
Eksperiment 2	1475	1842
Eksperiment 3	1684	N/A
Prosek	1494	1885

gOuroboros je u ovim testiranjima pokazao izvesnu dozu nestabilnosti te je bio neuspešan u većini slučajeva pri pokretanju testova. Ipak u dva slučaja kada su testovi uspešno završeni ukupan broj uspešnih transakcija bio je 1928 i 1842 respektivno. Prosek ova dva rezultata je 1885 uspešnih transakcija. U oba slučaja nije došlo do pojave neuspešnih transakcija ali je ova nestabilnost dovela do prekida testova u njegovoj početnoj fazi.

Ogmios je u ovim testovima bio stabilniji i testovi su uvek uspešno završeni. U tri pokrenuta testa rezultati su bili 1323, 1475 i 1684 uspešne transakcije respektivno, što daje prosek od 1494 uspešne transakcije.

Kako je Ogmios bio jako uspešan u svim pokretanjima sa 75 virtuelnih korisnika sprovedena su još tri testa sa po 100 virtuelnih korisnika. Ni u ovim testovima nije došlo do neuspešnih pokretanja te su tri pokretanja dala rezultate od 1858, 1765 i 1879 uspešnih transakcija respektivno, što daje prosek od 1834 uspešnih transakcija. Ni u jednom od pokrenutih testova sa 75 ili 100 virtuelnih korisnika nije došlo do pojave neuspešnih transakcija.

5. ZAKLJUČAK

Iz testiranja koje je sprovedeno za potrebe ovog rada može se zaključiti da je trenutno najsigurnije rešenje za kreiranje i slanje transakcija na Cardano blokchain mreže Ogmios jer pruža solidne performanse uz visoku pouzdanost u odnosu na druge spominjane alate. Veća propusnost gOuroboros biblioteke dolazi sa manjom otpornošću na greške, te trenutno može biti razmatran za korišćenje u nekritičnim, srednje opterećenim aplikacijama. Bitno je napomenuti da su greške dobijene testiranjem bile vezane za nedostupnost soketa sa kojim je gOuroboros biblioteka komunicirala te se može naslutitit da je postojeće rešenje moguće nadograditi.

Očekivano CLI je imao najslabije rezultate jer njegova namena nije zamišljena za slučajeve korišćenja koji su testirani u ovom radu. Pored toga, testiranje je pokazalo da je za mala opterećenja i ovo rešenje moguće koristiti bez većih problema i sa visokom sigurnošću ali da je skaliranje ograničeno.

6. LITERATURA

[1] A. Oram, „Harnessing the Power of Disruptive Technologies“, O'Reilly Media, pp. 8-15, 2001.

[2] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system”, 2008.

[3] R. Houben, A. Snyersm J. Cujkova *“*Cryptocurrencies and blokchain*”*, European Parliament, PE 619.024, Jul 2018.

[4] L. Arthur Ley, “Ultimate Cardano Smart Contracts – Unlock the Full Potential of the Cardano Blockchain by Developing Real-World Web 3.0,” Orange Education Pvt. Ltd., Jun 2024.

[5] https://ogmios.dev (pristupljeno u junu 2024.)

[6] https://grafana.com/docs/k6/latest/ (pristupljeno u junu 2024.)

Kratka biografija:

Miloš Maksimović rođen je u Kragujevcu 1999. god. Osnovne akademske studije na Fakultetu tehničkih nauka Univerziteta u Novom Sadu upisao je 2018. godine. Diplomirao je 2022. godine i iste godine upisao master akademske studije.
Kontakt: milos.maksimovic10@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.41
DOI: https://doi.org/10.24867/31BE18Rapic

MIGRACIJA MIKROSERVISNE ARHITEKTURE NA BEZSERVERSKO OKRUŽENJE UZ KORIŠĆENJE AWS SERVISA

MIGRATION FROM MICROSERVICE ARCHITECTURE TO A SERVERLESS PLATFORM USING AWS SERVICES

Marko Rapić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Ovaj rad istražuje prelazak sa mikroservisne arhitekture na bezserversku infrastrukturu u cloud okruženju. Implementacija koristi AWS servise, uključujući API Gateway, DynamoDB i Lambda funkcije, radi poboljšanja performansi i smanjenja troškova. Nova arhitektura omogućava efikasnije upravljanje sistemom i integrisano praćenje putem alata za monitoring.*

Ključne reči: *Bezserversko okruženje, Mikroservisi, AWS*

Abstract – *This paper explores the transition from microservices architecture to serverless infrastructure in a cloud environment. The implementation utilizes AWS services, including API Gateway, DynamoDB and Lambda functions, to improve performance and reduce costs. The new architecture enables more efficient system management and integrated monitoring.*

Keywords: *Serverless environment, Microservices, AWS*

1. UVOD

Sa sve većim zahtevima za skalabilnošću i optimizacijom resursa, bezserverska arhitektura postaje dominantan trend u *Cloud* okruženju. Ovaj pristup omogućava kompanijama da se fokusiraju na razvoj poslovne logike, dok infrastruktura i upravljanje resursima ostaju u nadležnosti *Cloud* platforme.

Rad istražuje proces migracije mikroservisne aplikacije na bezserversku arhitekturu, fokusirajući se na tehničke izazove, najbolje prakse i korake neophodne za uspešnu implementaciju. Migrirana aplikacija prvobitno je razvijena kao *Freelance* platforma za povezivanje slobodnih radnika i poslodavaca, omogućavajući kreiranje i upravljanje projektima, kao i praćenje toka rada.

Rešenje je zasnovano na korišćenjem *Amazon Web Services* (*AWS*) platforme, s posebnim fokusom na *AWS Lambda* funkcije koje omogućavaju izvršavanje bezserverskog koda u oblaku. Ove funkcije su razvijene u *.NET* ekosistemu, koristeći *C#* programski jezik, čime je obezbeđena kompatibilnost sa postojećom bazom koda originalne aplikacije.

Jedinstvenost ovakvog rešenja ogleda se u sposobnosti da iskoristi prednosti bezserverske infrastrukture uz zadržavanje glavnih osobina mikroservisne arhitekture.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Miroslav Zarić, redovni profesor

2. KORIŠĆENE TEHNOLOGIJE

Kao što je već pomenuto, rešenje se zasniva na korišćenju *AWS* platforme i njenih servisa. U ovom poglavlju biće opisani ključni servisi koji su korišćeni za uspešnu implementaciju aplikacije u bezserverskom okruženju.

2.1. *AWS Lambda*

AWS Lambda [1] je servis koji omogućava pokretanje koda bez potrebe za upravljanjem serverima. Ovaj bezserverski pristup znači da se može pokrenuti kod kao odgovor na događaje kao što su promene podataka, *HTTP* zahtevi, ili aktivnosti u drugim *AWS* servisima. *Lambda* automatski skalira aplikaciju pokretanjem koda kao odgovor na svaki okidač, bez obzira na obim saobraćaja.

2.2. *AWS DynamoDB*

DynamoDB [2] je brza, fleksibilna *NoSQL* baza podataka u potpunosti upravljana od strane *AWS*-a. Kao bezserverska baza podataka, ona automatski upravlja resursima, skaliranjem i dostupnošću, omogućavajući programerima da se fokusiraju na razvoj aplikacija bez brige o infrastrukturi. *DynamoDB* podržava *DynamoDB Streams*, što omogućava hvatanje i skladištenje svih promena koje se dešavaju u tabeli, uključujući operacije dodavanja, ažuriranja i brisanja podataka.

2.3. *AWS API Gateway*

AWS Api Gateway [3] servis služi za kreiranje, objavljivanje, održavanje i osiguranje *API*-a. *API Gateway* deluje kao "prednja vrata" za aplikacije, omogućavajući im pristup podacima, poslovnoj logici ili funkcionalnostima iz *backend* servisa. Ovaj servis podržava kreiranje *RESTful* i *WebScoket API*-a, omogućavajući i podršku za realnovremenu dvosmernu komunikaciju.

2.4. *AWS Cognito*

AWS Cognito [4] je servis za upravljanje autentifikacijom i autorizacijom korisnika u veb aplikacijama. Ovaj servis omogućava programerima da lako dodaju registraciju, prijavljivanje i pristupne kontrole u aplikacije. Njegova glavna prednost je jednostavna integracija sa ostalim *AWS* servisima radi poboljšanja sigurnosti i upravljanja korisničkim pristupom.

2.5. *AWS CloudWatch*

AWS CloudWatch [5] je servis za nadgledanje i upravljanje resursima na *AWS*-u. Omogućava prikupljanje, praćenje i

analizu metrike, logova i događaja sa različitih *AWS* servisa i aplikacija u realnom vremenu. *CloudWatch* je ključan za upravljanje i optimizaciju *AWS* okruženja, jer omogućava uvid u stanje i performanse.

2.6. *AWS EventBridge*

AWS EventBridge [6] je servis za upravljanje događajima u realnom vremenu, koji omogućava aplikacijama da se lako integrišu i reaguju na događaje iz različitih izvora. *EventBridge* omogućava izgradnju aplikacija koje odmah reaguju na promene u sistemu putem događaja.

3. SPECIFIKACIJA ARHITEKTURE SISTEMA

U specifikaciji fokus je na nefunkcionalnim zahtevima i arhitekturi sistema nakon migracije. Funkcionalni zahtevi same aplikacije nisu posebno razmatrani, jer se radi o prenetom rešenju gde su oni ostali u potpunosti nepromenjeni.

3.1. Nefunkcionalni zahtevi

U implementaciji nefunkcionalni zahtevi su od ključnog značaja za efikasnost i pouzdanost sistema. Fokus je stavljen na sledeće aspekte:

- Migracija na bezserversku arhitekturu putem *AWS* servisa, omogućavajući skalabilnost i minimiziranje infrastrukturnog održavanja
- *DevOps Pipeline* za *ASP.NET* servis koji ostaje van bezserverskog okruženja, uz automatski *CI-CD* proces za izgradnju, testiranje i isporuku
- Monitoring i logovanje preko *AWS CloudWatch* servisa, obezbeđujući uvid u performanse, stabilnost i sigurnost sistema

3.1. Arhitektura sistema

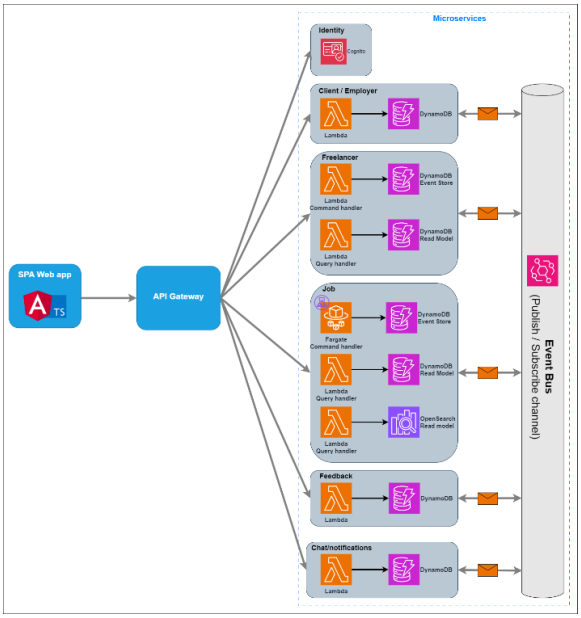
Posmatrajući arhitekturu sistema nakon migracije, ključna razlika u odnosu na originalno rešenje leži u prelasku na bezserversku infrastrukturu. Iako su funkcionalnosti postojećih komponenti ostale nepromenjene, sistem sada koristi *AWS* servise za efikasnije upravljenje resursima i skalabilnost. Na slici 1. prikazana je arhitektura nakon migracije, a nakon slike sledi opis ključnih komponenti sistema.

Komponenta „Identitet i pristup“ (*Identity*) predstavlja centralnu komponentu za upravljanje identitetom i pravom pristupa korisnika i igra ključnu ulogu u osiguravanju autentikacije i autorizacije sistema.

Komponenta „Profil klijenta“ (*Client/Employer*) zadužena je za upravljanje profilima klijenata.

Komponenta „Profil slobodnog radnika“ (*Freelancer*) ima ulogu upravljanja profilima slobodnih radnika unutar sistema. Ova komponenta je odgovorna za igradnju i upravljanje ličnim profilima slobodnih radnika, omogućavajući manipulaciju svim elementima profila.

Komponenta „Posao“ (*Job*) ima za cilj upravljanje celokupnim životnim ciklusom posla na Freelance platformi. Ova komponenta omogućava klijentima da objavljuju poslove i slobodnim radnicima da šalju predloge, što dalje može rezultovati uspešno sklopljenom poslu. Iz navedenog može se zaključiti da ova komponenta ima centralnu ulogu u povezivanju slobodnih radnika i klijenata.



Slika 1. Arhitektura sistema nakon migracije

Komponenta „Povratna informacija“ (*Feedback*) posvećena je dobijanju i deljenju ocena i komentara između klijenata i slobodnih radnika. Ova komponenta omogućava učesnicima obostrano učenje i podelu iskustva.

Komponenta „Notifikacije i chat“ (*Chat/Notifications*) omogućava komunikaciju (između klijenata i slobodnih radnika) i notifikacije u realnom vremenu. Chat funkcionalnost omogućava direktan dijalog i razmenu informacija, dok notifikacije informišu korisnike o važnim događajima na platformi.

EventBus je komponenta koja omogućava komunikaciju između različitih komponenti sistema putem definisanih integracionih događaja. Koristi se za prosleđivanje informacija o događajima bez direktne sprege između komponenti.

API Gataway je komponenta koja predstavlja sloj koji pruža jednostavan i ujedinjen interfejs za klijentske aplikacije. Ona obezbeđuje mapiranje i agregaciju podataka sa više različitih servisa, čime se smanjuje broj zahteva klijenta ka serverskoj strani i olakšava upravljanje.

SPA veb aplikacija je komponenta zadužena za interakciju sa korisnikom.

3. IMPLEMENTACIJA

U ovom poglavlju biće opisana implementacija ključnih servisa *Freelance* platforme nakon migracija na bezserversku infrastrukturu. U opisima implementacija fokus će biti na tehnološkim promenama. Takođe, biće objašnjena interna komunikacija servisa korišćenjem *AWS EventBridge*-a, a na kraju će biti predstavljeno rešenje za monitoring i upravljanje logovima pomoću *AWS CloudWatch*-a.

3.1. Mikroservisi

Tokom procesa migracije, mikroservisi su preneti sa *ASP.NET* radnog okvira na *AWS Lambda* servise, uz korišćenje *DynamoDB*-a za skladištenje podataka. Svaki od mikroservisa zadržao je svoje osnovne funkcionalnosti, ali je u okviru nove arhitekture prilagođen za rad u bezserverskom okruženju.

3.1.1 Identitet i pravo pristupa

U novoj arhitekturi sistema, servis za identitet i pravo pristupa migriran je na *AWS Cognito* servis.

Cognito obezbeđuje gotovo rešenje za upravljanje korisnicima, autentifikaciju i autorizaciju. Role u sistemu sada su definisane kroz *Cognito* grupe, koje omogućavaju razlikovanje između slobodnih radnika i poslodavaca, putem grupa *Freelancer* i *Employer*.

Za prijavu korisnika koristi se *Cognito Hosted UI*, koji nudi unapred pripremljen interfejs za prijavu na sistem, dok je registracija implementirana kao *AWS Lambda* funkcija. Ova *Lambda* funkcija ima zadatak da registruje korisnika u *Cognito* sistemu i doda ga u određenu grupu. Takođe, kreira i domenskog korisnika, bilo da je reč o slobodnom radniku ili poslodavcu.

3.1.2 Upravljanje profilima slobodnih radnika

Originalna implementacija mikroservisa bila je bazirana na *Clean* arhitekturi [7], koja je značajno olakšala proces migracije. *Clean* arhitektura podrazumeva jasno odvajanje logike sistema u nezavisne slojeve, što je omogućilo da se tokom migracije fokusiramo uglavnom na prezentacioni sloj. Konkretno, bilo je potrebno migrirati kod iz svakog kontrolera u *Lambda* funkciju, čime je obezbeđena besprekorna tranzicija ka bezserverskom okruženju.

Pored toga, u mikroservisu već je bio primenjen *CQRS* [8] (*Command Query Responsibility Segregation*) šablon, koji omogućava odvajanje operacija čitanja i pisanja u različite module. Primena *CQRS* šablona u sistemu omogućila je odvajanje modela pisanja i čitanja. Model pisanja predstavlja niz događaja koji se čuvaju u *DynamoDB* tabeli koja predstavlja skadište događaja (engl. *event store*). Za svaki zapisani događaj, podešen je *DynamoDB Stream* koji prati promene u tabeli i aktivira određene *Lambda* funkcije kada dođe do promene. Model čitanja predstavlja agregirane podatke o slobodnim radnicima, koji se čuvaju u posebnoj *DynamoDB* tabeli.

Ovaj *DynamoDB Stream* mehanizam omogućava automatsko obrađivanje događaja i njihovo korišćenje za sinhronizaciju podataka između modela čitanja i pisanja, čime se obezbeđuje konzistentnost i ažuriranost sistema.

3.1.3 Upravljanje poslovima

Poput prethodno opisanog servisa za upravljanje profilima slobodnih radnika, i ovaj servis je zasnovan na čistoj arhitekturi (*Clean Architecture*) i koristi *CQRS* šablon, ali je ovde primenjeno hibridno rešenje. Dok su upiti preobraženi u bezserversko okruženje, komande su zadržane u originalnom *ASP.NET* servisu.

Ovde fokus neće biti na *Lambda* funkcijama koje obrađuju model čitanja, jer im je implementacija identična kao u prethodnom servisu. Umesto toga, pažnja će biti usmerena na dva ključna aspekta: *DevOps* ciklus za *ASP.NET* servis i proces *deploy*-a ovog servisa na *AWS* infrastrukturu.

Prvi *DevOps* ciklus automatski proverava izgradnju (engl. *build*), prolaznost testova i *SonarCloud* [9] skeniranje koda koristeći *GitHub* akcije [10]. Drugi *DevOps* ciklus, koji se inicira ručno, radi *deploy* mikroservisa na *AWS* infrastrukturu.

Osnovu infrastrukture čini *AWS ECS* [11] koji omogućava upravljanje *Docker* [12] kontejnerima. U ovom slučaju, koristi se *AWS Fargate* [13], koji omogućava da se kontejneri izvršavaju bez potrebe za direktnim upravljanjem serverima. *Fargate* samostalno određuje i skalira potrebne resurse za svaki task, što značajno pojednostavljuje upravljanje infrastrukturom. Svaki task u *ECS*-u je definisan putem *ECS task definition*-a, koji opisuje sve potrebne parametre za pokretanje kontejnera, uključujući *Docker* sliku, mrežne portove, resurse kao što su memorija i procesor, i ostale važne konfiguracije.

Ovaj servis dobio je novu funkcionalnost koja omogućava korisnicima naprednu pretragu poslova koristeći *AWS OpenSearch* [15]. Ova pretraga koristi *OpenSearch* kao skladište podataka, dok se ažuriranje podataka automatizuje putem *Lambda* funkcija koje se aktiviraju na osnovu događaja poput kreiranja, ažuriranja ili brisanja poslova u modelu pisanja. Prilikom zahteva za pretragu, *Lambda* funkcija šalje upit *OpenSearch*-u i vraća relevantne rezultate, čime je omogućen brz i efikasan pristup poslovima u sistemu.

3.1.4 *Chat* i notifikacije

Implementacija *chat* sistema i notifikacija zahtevala je rešenje koje omogućava komunikaciju u realnom vremenu. S obzirom na potrebu za migracijom ovih servisa u bezserversko okruženje, kao idealno rešenje izabran je *AWS Api Gateway* u kombinaciji sa *WebSocket* protokolom.

U okviru *WebSocket API Gateway*-a, postoje podrazumevane rute *$connect* i *$disconnect* koje se automatski aktiviraju pri priključenju i prekidu konekcije korisnika. U slučaju ovog servisa, kada se korisnik priključi na *WebSocket*, ruta *$connect* aktivira *Lambda* funkciju koja čuva *connectionId* i *sub* (*JWT* token) korisnika u *DynamoDB* tabeli. Pri prekidu konekcije, ruta *$disconnect* aktivira funkciju koja uklanja ove podatke iz iste tabele. Ruta *sendMessage*, koja je povezana sa odgovarajućom *Lambda* funkcijom, omogućava slanje poruka izmedju korisnika uz pomoć podataka koji se čuvaju u prethodno pomenutoj tabeli.

3.1.5 Povratne informacije

Servis “Povratne informacije” predstavlja relativno jednostavan mikroservis. U procesu migracije u bezserversko okruženje, njegova logika je podeljena na više komandi, pri čemu je svaka komanda povezana sa odgovarajućom *Lambda* funkcijom. Podaci o povratnim informacijama čuvaju se u posebnoj *DynamoDB* tabeli, namenjenoj isključivo za tu svrhu.

Ključni aspekt servisa je obrada domenskog događaja završetka ugovora, koji okida *Lambda* funkciju koja kreira zapis u *DynamoDB* tabeli kako bi pripremila prostor za budući unos povratnih informacija.

3.2 Komunikacija između servisa

Nakon migracije sistema u bezserversko okruženje, pitanje komunikacije između mikroservisa ostalo je podjednako važno kao i u originalnoj arhitekturi.

3.2.1 *AWS EventBridge*

U procesu migracije, za međusobnu komunikaciju putem događaja korišćen je *AWS EventBridge.* U ovom sistemu, događaji stižu iz *DynamoDB Stream*-ova, ali pre nego što budu poslati na *EventBridge*, prolaze kroz posredničku *Lambda* funkciju. Ova funkcija, koja deluje kao dispečer, transformiše događaje u praktičnu strukturu, budući da su izvorni događaji iz *DynamoDB Stream*-a često neprikladni za direktnu obradu u drugim servisima. Tek nakon te transformacije, događaj se šalje na *EventBridge*, gde ga drugi servisi mogu presresti i obraditi.

3.2.2 *AWS API Gateway*

AWS API Gateway služi kao centralna tačka za upravljanje i usmeravanje *HTTP* zahteva u bezserverskom okruženju. U ovoj arhitekturi, *API Gateway* definiše sve rute koje vode do odgovarajućih *Lambda* funkcija, gde svaka funkcija obrađuje konkretan *HTTP* zahtev. Ovaj šablon obezbeđuje fleksibilnost i kontrolu nad komunikacijom između klijentskih aplikacija i mikroservisa, omogućavajući lako upravljanje i monitoring *API* poziva.

U originalnoj *ASP.NET* aplikaciji, *API Gateway* nije služio samo kao posrednik između klijenta i servisa, već je takođe igrao ulogu *API Composer*-a. To je značilo da je *API Gateway*, za neke rute, agregirao podatke iz više servisa i dostavljao ih klijentima kao jedinstven odgovor. Ova funkcija u bezserverskom okruženju implementirana je kroz specijalizovane *Lambda* funkcije. Ove *Lambda* funkcije su odgovorne za skupljanje i agregiranje podataka iz različitih mikroservisa.

3.3 Monitoring – *AWS CloudWatch*

U sklopu migracije, *AWS CloudWatch* se pokazao kao ključni alat za monitoring. Sa svim servisima koji su raspoređeni na *AWS*-u, *CloudWatch* sam po sebi obezbeđuje sveobuhvatno rešenje za praćenje performansi i analizu metrika.

Za svaki od korišćenih *AWS* servisa – kao što su *API Gateway*, *Cognito*, *DynamoDB* i *Lambda* funkcije – *CloudWatch* automatski generiše specifične panele koji predstavljaju vitalne metrike. Ovo je znatno ubrzalo dobijenje uvida u zdravlje i performanse sistema bez potrebe za dodatnim konfiguracijama.

4. ZAKLJUČAK

U ovom radu analiziran je proces migracije postojećeg sistema sa mikroservisne arhitekture na bezserversku infrastrukturu, sa ciljem povećanja skalabilnosti, smanjenja troškova i optimizacije upravljanja resursima. Mikroservisna arhitektura, iako veoma efikasna u radu sa kompleksnim sistemima, sa porastom broja korisnika donosi izazove u održavanju infrastrukture i upravljanju resursima. Ovi izazovi doveli su do potrebe za prelaskom na fleksibilniju arhitekturu koja omogućava lakše prilagođavanje rastućim zahtevima tržišta.

Migracija na bezserversku infrastrukturu baziranu na *AWS* uslugama, kao što su *Lambda* funkcije, *API Gateway* i *DynamoDB*, omogućila je automatsko skaliranje resursa i značajno smanjenje operativnih troškova. Ova arhitektura je pojednostavila upravljanje sistemom, jer *Lambda* funkcije reaguju isključivo na događaje, što eliminiše potrebu za stalnim upravljanjem serverima.

Postavljanjem infrastrukture na *AWS* otvorene su mogućnosti za dalje unapređenje sistema kroz integraciju sa drugim *AWS* uslugama. Na primer, primena *AWS OpenSearch*-a omogućila bi obradu velikih količina podataka u realnom vremenu, dok bi korišćenje mašinskog učenja i veštačke inteligencije moglo doprineti poboljšanju performansi i personalizaciji usluga kroz analizu podataka i predviđanje korisničkih potreba.

5. LITERATURA

[1] *AWS Lambda* servis za pokretanje koda u oblaku u bezserverskom okruženju https://aws.amazon.com/lambda/

[2] *AWS DynamoDB* bezserverska *NoSQL* baza podataka https://aws.amazon.com/dynamodb/

[3] *AWS API Gateway* servis za kreiranje ulazne tačke za aplikacije https://aws.amazon.com/api-gateway/

[4] *AWS Cognito* servis za autentifikaciju i autorizaciju korisnika https://aws.amazon.com/cognito/

[5] *AWS CloudWatch* servis za nadgledanje i upravljanje resursima https://aws.amazon.com/cloudwatch/

[6] *AWS EventBridge* servis za upravljanje dogadjajima u realnom vremenu https://aws.amazon.com/eventbridge/

[7] *Clean* arhitektura https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html

[8] *CQRS* šablon https://martinfowler.com/bliki/CQRS.html?ref=blog.funda.nl

[9] *SonarCloud Scan https://github.com/marketplace/actions/sonarcloud-scan*

[10] *GitHub Actions* https://github.com/features/actions

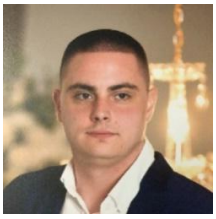
[11] *AWS ECS* servis za orkestraciju kontejnera https://aws.amazon.com/ecs/

[12] *Docker https://www.docker.com/*

[13] *AWS Fargate* servis za beserversko pokretanje kontejnera https://aws.amazon.com/fargate/

[15] *AWS OpenSearch* servis za pretraživanje i analizu velikih količina podataka https://aws.amazon.com/opensearch-service/

Kratka biografija:



Marko Rapić rođen je 26.02.2000. godine u Novom Sadu. Godine 2019. upisao je Fakultet Tehničkih Nauka u Novom sadu, odsek Računarstvo i automatika. Osnovne studije završio je u septembru 2023. Od oktobra 2023. upisuje Master akademske studije.
kontakt: rapic.marko26@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.9
DOI: https://doi.org/10.24867/31BE19Savic

MIGRACIJA MONOLITNE ARHITEKTURE NA BEZSERVERSKO OKRUŽENJE UZ KORIŠĆENJE AZURE SERVISA

MIGRATION FROM MONOLITE ARCHITECTURE TO A SERVERLESS PLATFORM USING AZURE SERVICES

Mihajlo Savić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu opisana je migracija monolitne arhitekture na bezserversku infrastrukturu u cloud-u. U cilju omogućavanja efikasnog monitoringa, smanjenja troškova i povećane skalabilnost, implementirani su različiti Azure servisi, uključujući SQL Server, SQL Database, Function App.*

Ključne reči: *Monolitne, Cloud, Azure[7],*

Abstract – *In this paper, the migration from a monolithic architecture to a serverless infrastructure in the cloud is described. To enable efficient monitoring, reduce costs, and increase scalability, various Azure services have been implemented, including SQL Server, SQL Database, and Function App.*

Keywords: *Monolithic, Cloud, Azure[7]*

1. UVOD

Povećanjem infrastrukture i količine zahteva, odnosno potrebe za skalabilnošću i bržim odgovorima potrebna su nova rešenja, od kojih je trenutno najbolje bezserverska arhitektura. Ona omogućava korisniku kompletan fokus na poslovnu logiku, dok upravljanje resursima i infrastrukturom preuzima izabrani provajder.

U rad je uključeno istraživanje i pronalaženje najboljeg načina za migraciju postojeće monolitne arhitekture u bezserversku arhitekturu. Aplikacija predstavlja softver za podršku rada cvećare, kako bi se olakšalo kreiranje porudžbina i narudžbina, kao i praćenje rada zaposlenih.

Implementacija je omogućena korišćenjem Azure servisa, sa najvećim fokusom na Azure Function App, odnosno funkcije koje se nalaze i izvršavaju na cloud-u. Funkcije su pisane u .NET[2] okruženju, u C#[1] programskom jeziku i koriste SQL bazu podataka, koja se takođe nalazi na cloud-u.

Usled korišćenja studentske licence za Azure, pojedini servisi nisu implementirani (Azure Active Directory B2C, Azure Application Gateway...) te oni predstavljaju moguća unapređenja na plaćenim verzijama.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Miroslav Zarić, vanr.prof..2. KORIŠĆENE

2. TEHNOLOGIJE

U ovom poglavlju opisani su ključni servisi koji su korišćeni za uspešnu implementaciju aplikacije u bezserverskom okruženju, kao i tehnologije korišćene u izradi gotove aplikacije.

2.1. KORIŠĆENE TEHNOLOGIJE

2.1.1. Angular[4]

Angular[4] je popularni open-source front-end framework koji se koristi za izgradnju dinamičnih web aplikacija. Razvijen je od strane Google-a i omogućava programerima da kreiraju jednostrane aplikacije (SPA - Single Page Applications) koje pružaju brže i interaktivnije korisničko iskustvo.

2.1.2. .NET[2]

.NET[2] je sveobuhvatan razvojni okvir (framework) koji omogućava izradu raznovrsnih aplikacija, uključujući web, desktop, mobilne i cloud aplikacije. Razvijen je od strane Microsoft-a i podržava više programskih jezika, najčešće C#[1], F# i VB.NET[2]. U ovom slučaju korišćen je C#[1] programski jezik.

2.1.3. Azure SQL Database[5]

Azure SQL Database je upravljana usluga baze podataka u cloudu koju nudi Microsoft Azure. To je relacijska baza podataka koja se oslanja na SQL Server tehnologiju, omogućavajući korisnicima da lako kreiraju, upravljaju i skaliraju baze podataka bez potrebe za održavanjem fizičke infrastrukture.

2.1.4. Azure Function App[8]

Azure Function App je serverless usluga koju nudi Microsoft Azure, koja omogućava razvoj i pokretanje malih, nezavisnih delova koda poznatih kao "funkcije". Ove funkcije se pokreću kao odgovor na različite događaje, omogućavajući programerima da reše specifične zadatke bez potrebe za upravljanjem infrastrukturom.

2.2. MOGUĆA UNAPREĐENJA

Usled korišćenja studentske licence neki od servisa nisu implementirani, te će oni biti navedeni kao moguća unapređenja na plaćenim verzijama.

2.2.1. Azure Active Directory B2C

Azure Active Directory B2C (Business to Consumer) je identitetska i pristupna usluga koju nudi Microsoft Azure, dizajnirana posebno za omogućavanje registrovanih korisnika da se autentifikuju i pristupaju web i mobilnim aplikacijama. Ova usluga omogućava preduzećima da upravljaju korisničkim identitetima i pristupom na jednostavan i siguran način.

2.2.2. Azure Application Gateway

Ova usluga je namenjena upravljanju saobraćajem za web aplikacije. Uključuje funkcionalnosti kao što su automatsko skaliranje, balansiranje opterećenja, sigurnosne karakteristike (WAF - Web Application Firewall) i SSL terminaciju. Omogućava optimizaciju performansi i zaštitu aplikacija.

2.2.3. Azure Storage

Implementacija Azure Storage bi bila korisna u daljim razvojima aplikacije u kojima će u porudžbinama biti prisutna slika.

Azure Storage je širok spektar usluga za skladištenje podataka koje nudi Microsoft Azure. Ova platforma omogućava korisnicima da čuvaju i upravljaju podacima u cloudu na siguran, skalabilan i visoko dostupni način. Azure Storage podržava različite tipove podataka, uključujući strukturisane, polustrukturirane i nestrukturirane podatke.

3. SPECIFIKACIJA

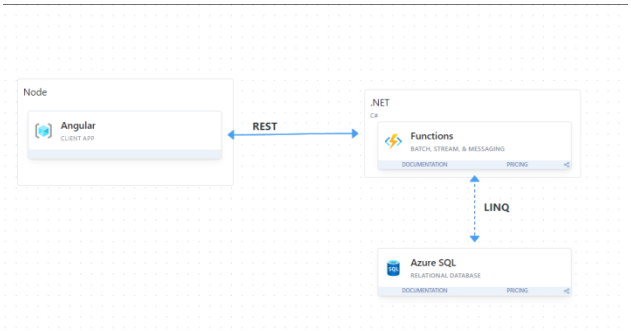
3.1. Nefunkcionalni zahtevi

U implementaciji fokus je stavljen na sledeće aspekte:

- Migracija na bezserversku arhitekturu putem *Azure* servisa, omogućavajući skalabilnost i minimiziranje infrastrukturnog održavanja
- *DevOps Pipeline* za *ASP.NET[2]* servis koji ostaje van bezserverskog okruženja, uz automatski *CI-CD* proces za izgradnju, testiranje i isporuku
- Monitoring i logovanje preko *Azure-a*, obezbeđujući uvid u performanse, stabilnost i sigurnost sistema

3.2. Arhitektura sistema

U ovom poglavlju detaljnije je opisana arhitektura sistema i tehnologije korišćene za razvoj istog. Softverski alat za podršku rada cvećare implementiran je kao veb-aplikacija koja se sastoji iz klijentskog i bezserverskog dela. Klijentski delovi navedene aplikacije implementirani su koristeći programski jezik TypeScript[3], a kao radni okvir odabran je Angular[4, 5], verzije 12. Centralni deo aplikacije implementiran je upotrebom bezserverskih tehnologija, funkcije su pisane u programskog jeziku C#[1] , korišćenjem radnog okvira .NET[2] verzije 6.0. Komunikacija u aplikaciji, vrši se preko HTTP protokola.



Node.js je neophodan za postavljanje razvojnog okruženja, upravljanje zavisnostima i olakšavanje procesa izgradnje i testiranja Angular[4] aplikacija. Iako se Angular[4] aplikacije izvršavaju na klijentskoj strani, Node.js pruža infrastrukturu i alate koji su ključni za njihov razvoj.

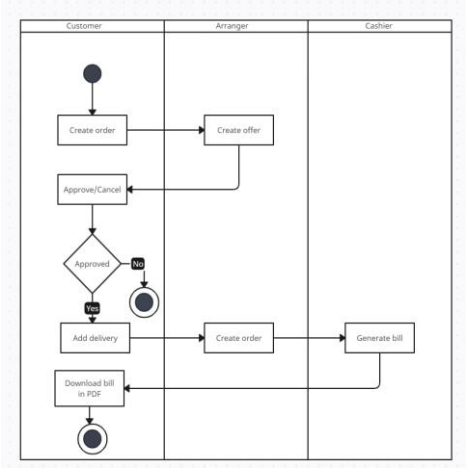
Angular[4] koristi HttpClient modul za komunikaciju sa backend-om putem RESTful API-ja. Zahtevi se šalju koristeći različite HTTP metode, a podaci se obično razmenjuju u JSON formatu. Korišćenje Observable-a omogućava asinkrono rukovanje podacima, dok RxJS operateri olakšavaju obradu odgovora i upravljanje greškama.

Azure Function App[8] omogućava brzo razvijanje i izvođenje funkcija koje se aktiviraju na osnovu različitih događaja. Kroz podršku za različite okidače, povezivanje sa drugim Azure servisima i mogućnost automatske skalabilnosti.

Azure SQL nudi fleksibilna i skalabilna rešenja za upravljanje relacionim bazama podataka u cloud-u. Sa visokom dostupnošću, sigurnosnim funkcijama, jednostavnom migracijom i snažnim alatima za monitoring.

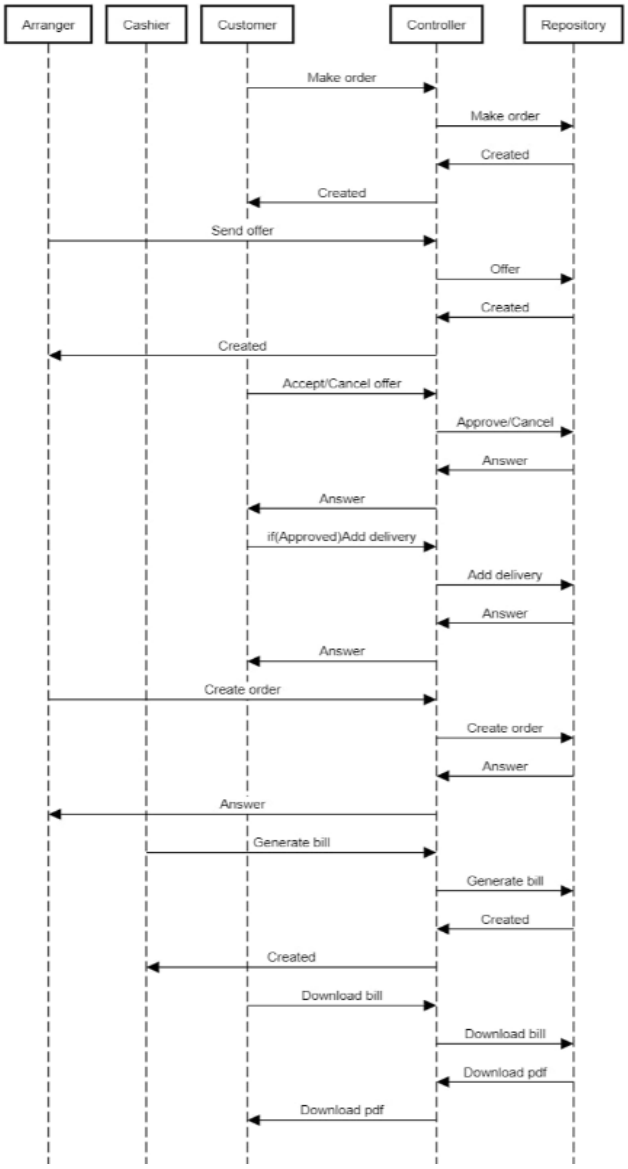
3.3. Activity diagram

Activity diagram je vrsta dijagrama u Unified Modeling Language (UML) koja se koristi za prikazivanje toka aktivnosti ili procesa unutar sistema. Ovi dijagrami su posebno korisni za modeliranje poslovnih procesa, radnih tokova i operacija u sistemu. Pomažu u razumevanju kako se aktivnosti međusobno povezuju i u kom redosledu se izvršavaju.



3.3. Sequence diagram

Sekvencni dijagram (sequence diagram) je jedan od najvažnijih dijagrama u Unified Modeling Language (UML) i koristi se za modeliranje interakcija između objekata ili komponenti u sistemu tokom vremena. Ovaj dijagram prikazuje kako objekti komuniciraju jedni s drugima putem slanja poruka, kao i redosled tih interakcija.



U datim dijagramima prikazana je komunikacija tri uloge unutar sistema (kupac, aranžer i kasir) kao i hronološko izvršavanje naredbi od strane svakog od njih.

Dijagrami su crtani u alatu Power Designer[6].

3.4. OPIS REALNOG SISTEMA

U ovom poglavlju opisano je pet klasa korisnika koje su od važnosti za ovaj softverski paket, kao i funkcionalni zahtevi koje ispunjava cvećara.

U realnom sistemu cvećare postoje sledeće klase korisnika koje su od interesa:

1. **Kupac** – poručuje aranžmane za koje može, a ne mora da zahteva dostavu.
2. **Aranžer** – pravi aranžmane kao i ponude na zahtevanu porudžbinu od strane kupca. Postavlja cene cveća i materijala.
3. **Šef** – njegova uloga sadrži mogućnost uvida u godišnju zaradu putem grafikona razdvojenog po mesecima i po godinama. Može dodeljivati povišice i smanjenja radnicima, naručiti robu od određenog dobavljača.
4. **Kasir** – ima mogućnost ispisivanja računa.
5. **Dobavljač** – izvršava dostave i ima uvid u sve dostave koje su poručene od strane poslovnica.

Karakteristike klasa korisnika informacionog sistema su:

1. **Neregistrovani korisnik i kupac** – poznavanje rada na računaru ovih uloga varira od lošeg do odličnog s obzirom da korisnici variraju od mlađih ka starijim, od edukovanih do needukovanih... Treba im omogućiti forme koje će posao raditi same, u smislu jednostavnih inputa ili dugmadi što je u našem sistemu i omogućeno.
2. **Aranžer** – očekuje se da osoba koja pripada ovoj grupi korisnika poseduje domensko znanje, kao i iskustvo aranžiranja u cvećari. Poznavanje rada na računaru spram informacionog sistema treba da bude poznato, s tim da se aranžer sa tim poslovima susretao i pre razvoja aplikacije. Ovoj grupi korisnika potrebno je što više olakšati posao jednostavnim formama za unos podataka u sistem, kako bi se više koncentrisali na razvoj proizvoda nego na ispravljanje bespotrebnih grešaka. Kako bi se sprečio nastanak grešaka pri unosu mnogobrojnih podataka u sistem, dosta pažnje je posvećeno validaciji podataka u toku unosa od strane prodavca.
3. **Šef** – pretpostavlja se da šef poseduje znanje rada na računaru, kao i domensko znanje, s obzirom da se bavi nabavkama i proračunima. Samim tim, nije nužno postavljanje jednostavnijeg korisničkog interfejsa, ali je poželjno da niko ne mora da se muči.
4. **Kasir i dobavljač** – za ove dve uloge se očekuje da imaju osrednje znanje rada na računaru, pogotovo kasir, kojoj je i posao računar (kasa). Omogućen im je jednostavan interfejs sa kojim ne bi trebali imati problema.

4. ZAKLJUČAK

U ovom radu analiziran je proces migracije postojećeg sistema sa monolitne arhitekture na bezserversku infrastrukturu, sa ciljem povećanja skalabilnosti, smanjenja troškova i optimizacije upravljanja resursima. Monolitna arhitektura, iako veoma jednostavna za razvoj, laka za testiranje I debagovanje sobzirom da su sve komponente unutar jednog paketa, omogućava direktnu komunikaciju, nailazi na probleme sa povećanjem broja korisnika. Neki od problema su održavanju infrastructure I upravljanju resursima. To je dovelo do prelaska na arhitekturu koja omogućava lakše prilagođavanje ovim problemima.

Migracija na Azure arhitekturu koja uključuje Azure funkcije kao i Azure SQL bazu podataka doprinela je poboljšavanju performansi, kao i donela mogućnost automatskog skaliranja aplikacije u situacijama kada to postaje potrebno.

Azure funkcije su funkcije s „hladnim pokretanjem“ (*cold start*) što takođe doprinosi upravljanju resursima, i kontroli troškova, jer se instance pokreću samo kada su potrebne. Ovo smanjuje troškove, jer korisnici plaćaju samo za vreme izvršenja. Prednosti im je takođe i fleksibilnost i skalabilnost jer omogućavaju lako skaliranje aplikacija prema potrebama korisnika bez potrebe za unapred definisanjem kapaciteta.

Korišćenje Azure arhitektura omogućava dalju intergraciju sa ostalim Azure servisima koji su navedeni iznad.

4. LITERATURA

[1] C# Language Documentation, https://learn.microsoft.com/en-us/dotnet/csharp/

[2] What is .NET, .NET Documentation, https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet

[3] TypeScript Documentation, https://www.typescriptlang.org/docs/

[4] Angular Documentation, https://v17.angular.io/docs

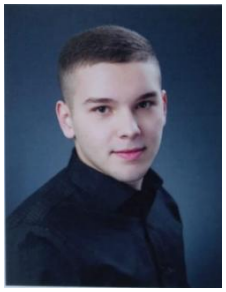
[5] Azure SQL Database, https://learn.microsoft.com/en-us/azure/azure-sql/database/sql-database-paas-overview?view=azuresql

[6] PowerDesigner, https://www.powerdesigner.biz/EN/powerdesigner/lp-data-modeling-powerdesigner-trial-source_adwdatamodel.html?gad_source=1&gbraid=0AAAAAocL3fkVU5cOzdN78FR0PPREp7_vG&gclid=Cj0KCQjw05i4BhDiARIsAB_2wfBNjhs4Khim-AHrd_nzc2--T2zSi4Jo1p9iqSTtUT5qP5WWM7ocd9kaAs8SEALw_wcB

[7] Azure Documentation, https://learn.microsoft.com/en-us/azure/?view=azuresql&product=popular

[8] Azure Functions Overview, https://learn.microsoft.com/en-us/azure/azure-functions/functions-overview?pivots=programming-language-csharp

Kratka biografija:



Mihajlo Savić rođen je 05.08.2000. godine u Novom Sadu. Godine 2019. upisao je Fakultet Tehničkih Nauka u Novom sadu, odsek Računarstvo i automatika. Osnovne studije završio je u septembru 2023. Od Oktobra 2023. upisuje Master akademske studije.
kontakt: justdoit0508 @gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.9
DOI: https://doi.org/10.24867/31BE21Mirilovic

GENERATOR KODA ZA MOBILNE APLIKACIJE

CODE GENERATOR FOR MOBILE APPLICATIONS

Olivera Mirilović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je predstavljen softverski alat vođen modelima za specifikaciju i generisanje koda mobilnih aplikacija. Rad obuhvata analizu postojećih rešenja koja koriste veštačku inteligenciju, low-code platforme i cross-platform radne okvire. Razvijeni meta-model i generator koda omogućavaju generisanje mobilnih aplikacija. Cilj ovog rada je da pokaže da implementirani generator može, do određene mere, omogućiti trenutnim ili budućim programerima brži razvoj mobilnih aplikacija, kao i da doprinese proširenju njihovog znanja i iskustva.*

Ključne reči: *Inženjerstvo vođeno modelima, MDE, meta-model, MagicDraw, generator, mobilne aplikacije*

Abstract – *This paper presents a model-driven software tool for the specification and code generation of mobile applications. It includes an analysis of existing solutions that utilize artificial intelligence, low-code platforms, and cross-platform frameworks. The developed meta-model and code generator enable the generation of mobile applications. The goal of this paper is to demonstrate that the implemented generator can, to a certain extent, facilitate the development of mobile applications for current and future developers, as well as contribute to the expansion of their knowledge and experience.*

Keywords: *Model driven engineering, MDE, meta-model, MagicDraw, generators, mobile applications*

1. UVOD

Generatori mobilnih aplikacija postali su sve istaknutiji poslednjih godina, nudeći pristupačna sredstva za preduzeća, pojedince i netehničke korisnike da kreiraju aplikacije bez potrebe za opsežnim znanjem kodiranja. Ove platforme su pojednostavile razvoj aplikacija obezbeđujući unapred izgrađene šablone, lako dostupne interfejse i prilagodive funkcije. Sve veća upotreba mobilnih uređaja i potreba za efikasnim rešenjima koja će zadovoljiti zahteve korisnika dodatno su podstakli rast ovih platformi. Značaj generatora mobilnih aplikacija leži u njihovoj sposobnosti da ubrzaju proces razvoja.

Predstavljeni softverski alat sastoji se od sledećih modula. Prvi modul sadrži metamodel na osnovu kojeg se specificira model mobilne aplikacije, dok drugi modul, generator, obuhvata kolekciju šablona za generisanje izvršivog programskog koda na osnovu dobijenog modela iz prvog modula. Osnovni motiv za razvoj generatora mobilnih aplikacija je ubrzanje procesa njihovog razvoja, kao i podsticanje širem krugu korisnika da kreiraju i održavaju mobilne aplikacije.

2. POSTOJEĆA REŠENJA

Neki od pristupa koji se koriste pri razvoju generatora mobilnih aplikacija uključuju cross-platform radne okvire, low-code platforme i veštačku inteligenciju. Među ovim pristupima, razvoj softvera vođen modelom privukao je pažnju zbog svog potencijala da automatizuje značajan deo procesa razvoja mobilnih aplikacija.

Jedna od istaknutih metoda je upotreba cross-platform radnih okvira koji omogućavaju programerima da pišu aplikacije na jednom jeziku koji se zatim može kompajlirati u kod specifičan za platformu. Radni okvir u ovoj kategoriji vredan pomena je Apache Cordova [1], koji omogućava razvoj mobilnih aplikacija koristeći HTML, CSS i JavaScript, a zatim ih umotava u izvorni kontejner. Prema ovom radu [2], ovaj pristup je značajno smanjio vreme i troškove razvoja mobilnih aplikacija omogućavajući da jedna baza koda funkcioniše na više platformi. I pored njegovih prednosti, postoje primetni problemi sa performansama, posebno u grafički intenzivnim aplikacijama koje mogu biti veoma ograničene.

Drugi pristup fokusira se na low-code platforme koje pružaju programerima mogućnost da kreiraju mobilne aplikacije putem grafičkih interfejsa i unapred izgrađenih komponenti. OutSystems [3] je dobar primer ovog pristupa, a njegove prednosti su istražene u radu [4]. Dok ove platforme čine razvoj pristupačnijim, one mogu ograničiti prilagodivost, jedinstvenost komponenti i mogu imati poteškoća da ispune zahteve performansi složenih aplikacija.

Napredak u okvirima veštačke inteligencije orijentisanim razvojnim okruženjima baziranim na modelima nedavno je došao u prvi plan razvoja mobilnih aplikacija. Istraživanje u radu [5] proučava kako metode vođene modelom, proširene veštačkom inteligencijom, mogu dalje optimizovati generisanje aplikacija sugerišući dizajn interfejsa i poboljšavajući efikasnost koda na osnovu unapred definisanih modela. Ova kombinacija MDE i veštačke inteligencije predstavlja oblast u razvoju koja obećava dalja smanjenja ručne intervencije u procesu kreiranja aplikacija.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Gordana Milosavljević, red. prof.

3. KORIŠĆENE TEHNIKE I TEHNOLOGIJE

MagicDraw alat je korišćen za dobijanje podataka o elementima modela, a Java aplikacija, sa FreeMarker obrađivačem šablona, za generisanje same aplikacije. Rezultirajuća mobilna aplikacija sastoji se od klijentske strane bazirane na React Native Expo i serverskog dela sa Spring i REST servisima.

3.1. MagicDraw

MagicDraw je alat za modelovanje koji se prvenstveno koristi za dizajn i arhitekturu softvera. Podržava jezike za modelovanje kao što je UML (Unified Modeling Language). Široko se koristi u razvoju softvera za projektovanje arhitekture sistema, generisanje koda i upravljanje zahtevima. Stereotip u MagicDraw predstavlja mehanizam za proširenje standardnog UML metamodela. Primenom stereotipa, jezik modeliranja može se prilagoditi tako da bolje odgovara specifičnim potrebama projekta, pružajući više konteksta ili dodatne atribute elementima u dijagramu.

3.2. Apache Freemarker

Apache Freemarker je obrađivač šablona, odnosno Java biblioteka za generisanje tekstualnog sadržaja na osnovu šablona i promenljivih podataka. Šabloni su napisani u FreeMarker Template Language jeziku, koji je jednostavan i specijalizovan. Obično se za pripremu koristi programski jezik opšte namene, kao što je Java, za izdavanje upita u bazi podataka i obavljanje poslovne logike. Nakon toga, Freemarker prikazuje pripremljene podatke koristeći šablone.

3.3. REST

Representational State Transfer (REST) predstavlja stil softverske arhitekture koji definiše skup smernica za kreiranje pouzdanih stateless veb API-ja. Veb API koji se pridržava REST ograničenja opisuje se kao RESTful API. RESTful API se oslanja na HTTP protokol, pa se operacije nad resursima svode na HTTP metode kao što su GET, POST, PUT, DELETE i OPTIONS.

3.4. Spring

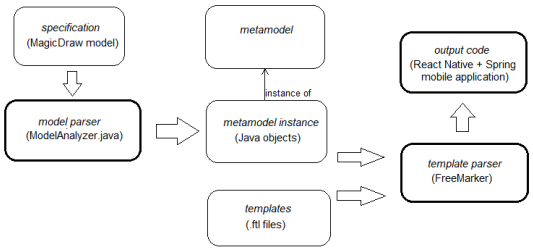
Spring je radni okvir napisan u Java programskom jeziku koji omogućava izgradnju REST arhitekture za Java aplikacije. Spring upravlja infrastrukturom tako da programer može da se fokusira na domenske probleme umesto na tehnologiju i realizaciju. U implementiranoj aplikaciji, paketi koji su primarno korišćeni sadržali su kontrolere, servise i repozitorijume.

3.5. React Native Expo

React Native je radni okvir razvijen od strane Facebook-a koji omogućava programerima da kreiraju mobilne aplikacije koristeći JavaScript i React. Za razliku od tradicionalnih pristupa razvoju mobilnih aplikacija, koji zahtevaju odvojene baze koda za iOS i Android, React Native omogućava programerima da napišu jednu bazu koda koja radi na obe platforme. Expo je radni okvir i platforma izgrađena oko React Native koja dodatno pojednostavljuje proces razvoja pružanjem alata i usluga koje pomažu programerima da brzo izgrade, primene i iteriraju svoje mobilne aplikacije. Uključuje paket kao što je Expo Go, mobilna aplikacija koja omogućava programerima da pregledaju svoje aplikacije na fizičkim uređajima bez potrebe za kompajliranjem koda.

4. ARHITEKTURA GENERATORA MOBILNIH APLIKACIJA

Odnos između MagicDraw alata i generatora koda u sistemu za generisanje mobilnih aplikacija može se videti na slici 1. Kao što je prikazano, izlaz iz MagicDraw alata je specificirani model i koristi se kao ulaz za analizator, koji ga mapira na elemente generatora koji se podudaraju sa elementima meta-modela. Izlaz iz analizatora su Java objekti koji će se, zajedno sa šablonima, koristiti kao ulaz za Apache FreeMarker obrađivač šablona, iz kojeg se dobija izlazni kod mobilne aplikacije.

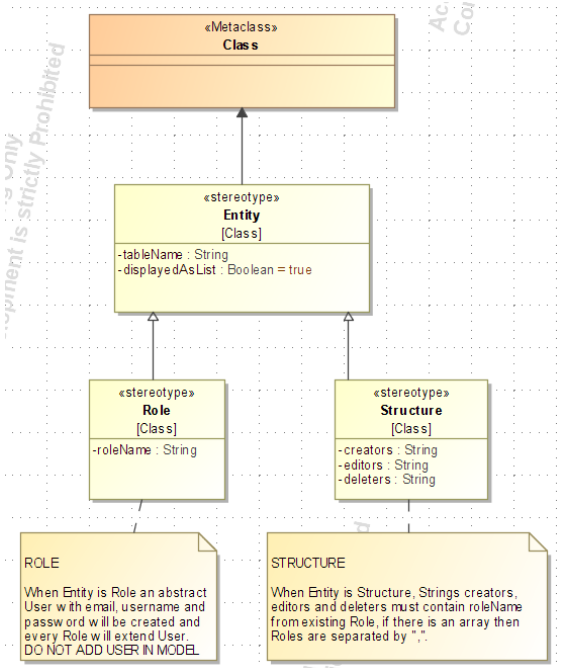


Slika 1 *Arhitektura generatora mobilnih aplikacija*

5. IMPLEMENTACIJA METAMODELA I MODELA

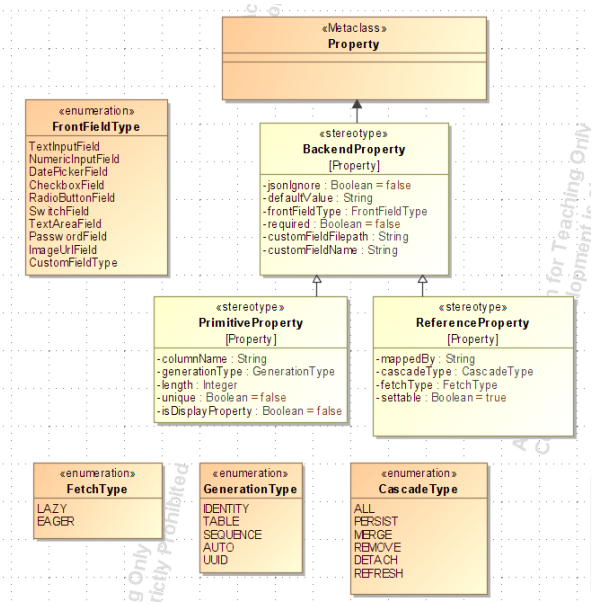
U kontekstu MDE, model je opis sistema ili dela sistema zapisan korišćenjem dobro definisanog jezika koji podržava automatsku interpretaciju od strane računara. Za kreiranje jezika primenjen je mehanizam koji koristi UML kao jezik opšte namene za prilagođavanje domenu primene. Proširenje UML-a može se sprovesti na više načina, a specijalizacija koja je primenjena u ovom radu je proširenje putem definisanja UML profila. Ova metoda podrazumeva definisanje stereotipa koji predstavljaju proširenje postojećih metaklasa, kao i definisanje označenih vrednosti, tagova koji predstavljaju meta-atribute stereotipa, i ograničenja koja mogu postavljati restrikcije na postojeća pravila jezika, ali ih ne mogu ukidati. Deo UML profila koji je korišćen za specifikaciju mobilne aplikacije može se videti na slici 2. U njemu su definisani stereotipi *Role* i *Structure*, koji nasleđuju *Entity* stereotip, koji predstavlja proširenje osnovne metaklase *Class*. Ako klasa predstavlja korisničku klasu, onda će se u modelu označiti sa *Role* stereotipom; u suprotnom, označiće se *Structure* stereotipom, u okviru kog se mogu definisati tipovi korisnika kojima je dozvoljeno kreiranje, uređivanje i brisanje te klase popunjavanjem tagova creators, editors i deleters. Ako ne postoje korisnici u modelu, te operacije su dostupne nad svim klasama koje su

prikazane, odnosno imaju *displayedAsList* tag označen sa true.



Slika 2 - *UML profil za proširenje metaklase Class stereotipima Entity, Role i Structure*

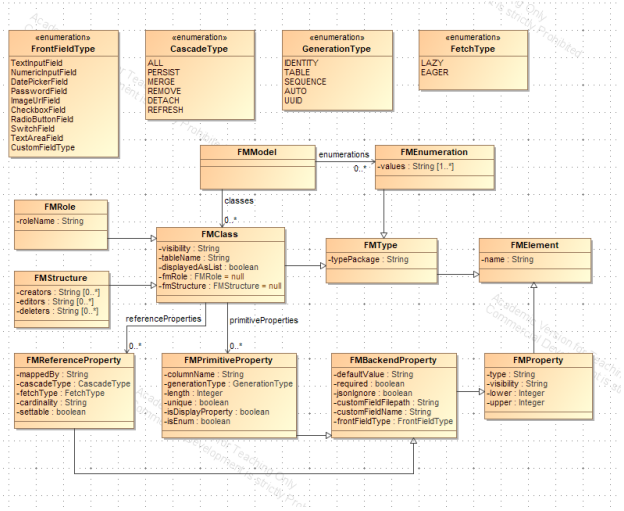
Na slici 3 je prikazan deo metamodela u kojem se proširuje mataklasa *Property* koja predstavlja apstrakciju promenljivih. Stereotipom *BackendProperty* se takodje definiše i tip polja koji će se koristiti u aplikaciji pri popunjavanju forme za kreiranje i uređivanje entiteta. Ovaj stereotip nasleđuje *PrimitiveProperty* stereotip, kojim se označavaju atributi koji su primitivnog tipa, i *ReferenceProperty*, kojim se označavaju atributi koji su tipa klase.



Slika 3 - *UML profil za proširenje metaklase Property stereotipima BackendProperty, PrimitiveProperty i ReferenceProperty*

6. IMPLEMENTACIJA GENERATORA KODA

Generator koda se najčešće implementira na bazi nekog obrađivača šablona. U ovom sistemu je razvijem korišćenjem Apache FreeMarker-a pomoću kog su kreirani šabloni za: klase, kontrolere, servise, repozitorijume, enumeracije, kao i za fajlove na klijentskoj strani. Model sistema za generisanje koda se sastoji od 11 klasa i 4 enumeracije (Slika 4). Struktura klasa odgovara elementima kreiranog UML profila kako bi se podaci o elementima modela mogli mapirati na elemente generatora.

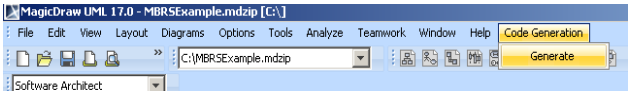


Slika 4 - *Dijagram klasa generatora za prihvat podataka iz modela*

Jedan od prvih koraka u radu generatora je analizirati dobijeni model aplikacije iz MagicDraw-a i mapirati sve podatke na strukturu podataka generatora optimizovanu za upotrebu od strane obrađivača šablona. Ova akcija se izvršava u klasi *ModelAnalyzer*. Analizator iterira kroz elemente paketa i izdvaja podatke koje skladišti u prethodno definisanim klasama i enumeracijama. Kada analizator završi svoj rad i svi podaci su namapirani na odgovarajuće strukture podataka, pokreće se generisanje koda. Ova funkcija je podeljena na više generatora koji su organizovani po vrsti fajla koji se generiše ili po podacima koji su potrebni u šablonima. Pri tome, svaki generator ima definisan naziv, lokaciju fajla u kojem se nalazi šablon za generisanje koda, naziv i tip fajla koji će biti izgenerisan, kao i lokaciju u aplikaciji gde će fajl biti sačuvan.

7. PRIKAZ RADA SISTEMA

Prvi korak u generisanju koda je kreiranje modela aplikacije u MagicDraw alatu i povezivanje sa postojećim meta-modelom kako bi se mogli koristiti stereotipi definisani unutar njega. Kada se to uradi, pokreće se Java projekat. Nakon toga automatski se pokreće MagicDraw alat u okviru kog će se pojaviti stavka u meniju sa nazivom *Code Generation* (Slika 5). Kada se klikne na *Generate* izgenerisaće se nov direktorijum u okviru Java projekta u okviru kog su dva podfoldera sa serverskom i klijentskom stranom mobilne aplikacije.



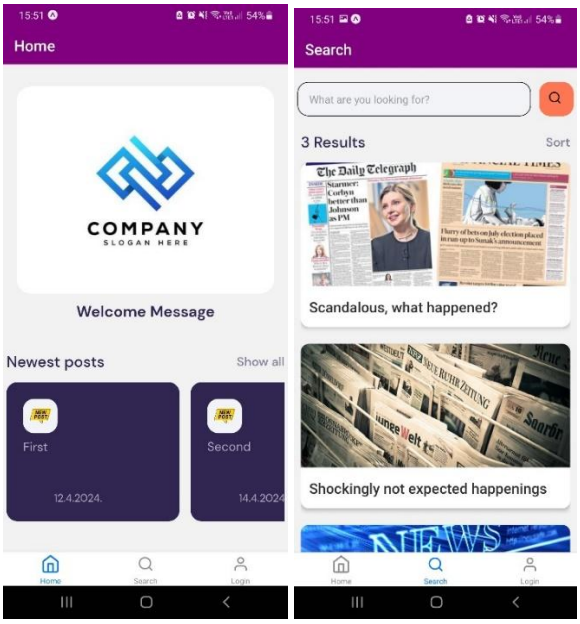
Slika 5 - *MagicDraw meni sa stavkom Code Generation*

Za pokretanje klijentske strane koristi se Expo Go mobilna aplikacija. Pošto je razvijena upotrebom React Native, omogućen je rad i na Android i na iOS operativnom

sistemu. Rad aplikacije će biti predstavljen korišćenjem aplikacije za blog objave. Sledeći tabovi su dostupni pri pokretanju mobilne aplikacije:

- *Home* tab (Slika 6(A)) – početni ekran koji se pojavljuje svim korisnicima, nezavisno od toga da li postoje korisničke uloge u modelu.
- *Login* tab – ekran za prijavu, postoji ukoliko su definisane korisničke uloge u modelu. Ispod prijave je moguće pristupiti stranici za registraciju klikom na Register.
- *Search* tab (Slika 6(B)) – ekran za pretragu nad listom, postoji nula ili više ukoliko se u modelu postavi meta-atribut *displayedAsList* na true nad klasom. U ovom primeru je samo nad *Post* klasom definisan *displayedAsList*. Klikom na jednu od kartica preusmerava se na nov prozor koji sadrži detalje o tom elementu.
- *Account* tab – nakon uspešne prijave *Login* tab se zamenjuje *Account* tabom. U okviru ovog prozora korisnik ima dugme da uredi svoje korisničke podatke, dugme za uređivanje podataka koje su jedinstveni po tipu korisnika, dugme za izmenu lozinke, dugme za brisanje naloga i dugme za logout.

Ukoliko postoje korisničke uloge u modelu, nakon logovanja će se u gornjem levom uglu ekrana pojaviti meni. U njemu se nalaze liste koje postoje u korisničkoj klasi. Takođe, tu su i liste objekata klase za koju je korisnička uloga definisana da može da kreira, uređuje ili briše, putem meta-atributa *creators*, *editors* i *deleters*. Ako ne postoje korisničke uloge u modelu, tada će *Search* tab sadržati dugmiće za brisanje objekta kartica i dugme za kreiranje novih objekata u donjem desnom ćošku ekrana. Kada se klikne na dugme, otvoriće se novi prozor sa formom koja sadrži sva polja definisana u modelu. Sva polja čiji meta-atribut *required* ima istinitu vrednost su označena zvezdicom u opisu, i ako se ne popune pre klika na *Save*, pojaviće se dijalog za upozorenje da nedostaje vrednost.



Slika 6 - *(A) Home tab i (B) Search tab*

8. ZAKLJUČAK

U ovom radu predstavljen je sistem za generisanje mobilnih aplikacija korišćenjem metoda inženjerstva softvera vođenog modelom. Glavna ideja inženjerstva vođenog modelom je da odvoji dizajn od tehničke implementacije, što programerima omogućava da se fokusiraju na koncepte iz domena problema, dok se automatski sistemi bave detaljima generisanja koda. Značaj ovog pristupa leži u tome što promoviše ponovnu upotrebljivost, skalabilnost i održivost sistema, kao i brži razvoj. U radu su prikazani motivacija za razvoj sistema, korišćene tehnike i tehnologije. Prikazana je implementacija meta-modela u alatu MagicDraw, kao i njegov značaj pri kreiranju modela. Predstavljen je rad generatora koda napisanog u Javi, sa analizatorom za mapiranje elemenata definisanih u modelu na elemente generatora, dok je FreeMarker korišćen kao obrađivač šablona. Budući rad može uključivati proširenje meta-modela za organizaciju komponenti i stranica u mobilnoj aplikaciji, kao i unapređenje tehnologija korišćenih u razvoju.

9. LITERATURA

[1] Apache Cordova, https://cordova.apache.org/, преузето септембра 2024.

[2] Charland, Andre, and Brian Leroux. "Mobile application development: web vs. native." *Communications of the ACM* 54.5 (2011): 49-53.

[3] OutSystems, https://www.outsystems.com/, преузето септембра 2024.

[4] Gomes, Pedro M., and Miguel A. Brito. "Low-code development platforms: a descriptive study." *2022 17th Iberian Conference on Information Systems and Technologies (CISTI)*. IEEE, 2022.

[5] Cervera, R. G., Esteban, D., & Garijo, F. J. (2010). "Model-driven development of mobile applications. " *Journal of Systems and Software*, *83*(10), 1715-1732.

Kratka biografija:

Olivera Milivojević rođena je u Novom Sadu 1999. godine. Diplomirala je na Fakultetu tehničkih nauka u Novom Sadu, na smeru Softversko inženjerstvo i informacione tehnologije 2022. godine. Master rad na Fakultetu tehničkih nauka iz oblasti Softverskog inženjerstva o informacionih tehnologija odbranila je 2024.god.

Зборник радова Факултета техничких наука, Нови Сад

UDK: 4.9
DOI: https://doi.org/10.24867/31BE22Lazarevic

АНАЛИТИЧКА ПЛАТФОРМА ЗА ПОДРШКУ ПРИСТУПУ РАЗВОЈА СОФТВЕРА ВОЂЕНОМ ПОДАЦИМА

ANALYTICS PLATFORM FOR THE SUPPORT OF DATA-DRIVEN APPROACH OF SOFTWARE DEVELOPMENT

Тамара Лазаревић, *Факултет техничких наука, Нови Сад*

Област – ПРИМЕЊЕНЕ РАЧУНАРСКЕ НАУКЕ И ИНФОРМАТИКА

Кратак садржај – *У овом раду представљена је имплементација, интеграција са примером апликације у оквиру модела софтвера као услуге (енг. SaaS), као и евалуација аналитичке платформе за подршку развоја софтвера вођеног знањем. Платформа је развијена у циљу повећања изгледа за успешан пласман производа и повећања конкурентске предности компанија на тржишту.. Пројектована је модуларно, при чему су дати конкретни алати за имплементацију наведених модула. Улаз у платформу представљају подаци који потичу из интеракције корисника са апликацијом, док је излаз скуп дешборда који омогућавају информисано доношење одлука о даљем развоју апликације. Резултати евалуације потврдили су да решење испуњава захтеве за реалну употребу.*

Кључне речи: *аналитике, платформа, data-driven, развој софтвера, рударење знања*

Abstract – *This paper presents the implementation, integration with an example application within the Software as a Service (SaaS) model, and the evaluation of an analytical platform for supporting data-driven software development. The platform aims to boost the chances of successful product launch and enhance companies' competitive advantage in the market. It is modular, with specific tools provided for implementing the stated modules. The platform's input consists of data derived from user interactions with the application, while the output is a set of dashboards that enable informed decision-making about the further development of the application. The evaluation results confirmed that the solution meets the requirements for real-world use.*

Keywords: *analytics, platform, data-driven, software development, data mining*

1. УВОД

Адекватна примена компанијског знања, знања о производу и корисницима кључна је за компетитивну предност компанија [1], [2].

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији ментор је био др Александар Ковачевић, ред. проф.

Истраживања истичу важност повезивања корисничког знања са инжењерским процесима за успешан пласман производа на тржиште [2].

Успех компанија у којима су одлуке вођене подацима (енг. *data-driven*) зависи од ефикасне примене знања на производе који доносе вредност корисницима [2], па кључан проблем представља одрживо и исплативо прикупљање и интеграција тог знања у развојне процесе.

У раду је дат предлог реализације платформе за подршку дата-дривен приступу развоја софтвера.

Примена платформе илустрована је на случају пример софтвер као услуга (енг. *SaaS, Software as a service*) апликације за организацију посла која прати канбан методологију, пружајући комплетан графички увид у пројекте.

Подаци који фигуришу у платформи представљају синтетичке податке који би настали потенцијалним корисничким коришћењем апликације.

Представљена је модуларна архитектура платформе, описана интеграција са постојећим компонентама апликационог система, као и крајњи излаз из платформе и његови бенефити. Дати су и конкретни алати који се могу користити за имплементацију модула платформе, и њихове предности.

Евалуација платформе обављена је тестирањем перформанси неколико модула под редовним и вишим интензитетом коришћења, употребом алата за тестирање система под оптерећењем. Добијени резултати указују на то да платформа може успешно да одговори на оба типа оптерећења и неометано обавља своју функционалност, и да моделовање података знатно утиче на перформансе платфоме.

2. ПРЕГЛЕД СТАЊА У ОБЛАСТИ

2.1. Примена знања о корисницима у процесу развоја софтвера

Ране студије релевантне за проблематику истичу значај управљања знањем у компанијама [1]. Касније истраживања се фокусирају и на важност разумевања потреба и понашања корисника за постизање конкурентске предности и финансијског успеха компанија [2].

У раду [3] истичу се предности коришћења техника машинског учења и рударења знања (енг. *data mining*) за компаније: за идентификацију првих корисника, фокусирања комуникације на препознати сегмент купаца, предвиђања будуће потражње и откривања оптималних побољшања за производ или услугу.

Радови из последњих 5 година потврђују значај знања о корисницима, и позивају на на интеграцију знања о корисницима у процес развоја производа [4]. Изучаван је и утицај дигитализације и растуће улоге података у процесу дизајна и развоја производа, с акцентом на академска питања која произилазе из тога. Ова питања обухватају откривање могућих концептуалних модела, процедура и платформи за прикупљање података [5], што је обрађено у раду.

2.2. Технике рударења знања за екстракцију знања о корисницима

У литератури која се бави употребом техника рударења знања за екстракцију знања о корисницима, коришћене технике рударења знања укључују асоцијативна правила, кластеровање, стабла одлучивања (енг. *decision trees*) , мајновање знања из текста (енг. *text mining*), конџоинт анализу (енг. *conjoint analysis*) и друге, и употребљавају се у различитим фазама развоја производа [4].

Извори података о корисницима у литератури су најчешће резултати интернет анкетирања [2], [4]. Резултујући скупови података могу представљати ограничен узорак са малом варијансом, што може утицати на биас резултата. Овај рад насупрот, предлаже методологију прикупљања знања директно од корисника, насталог њиховом употребом и интеракцијом са апликативним производом.

2.3. Системи за управљање подацима

Рад [6] дели архитектуре аналитичких платформи у три групе. Прва група обухвата архитектуре у којима складиште података има централну улогу, где су мане мањак подршке за неструктуиране податке и високе трошкове платформи за велике количине података. Модернији приступ су двослојне архитектуре које укључују слој језера података (енг. *data lake*). Подаци се чувају у слојевима са ниском ценом складиштења, а само подскуп података се трансформише и даље транспортује у складишта података за аналитичке примене. Према раду, користе се у готово свим Форчн 500 компанијама. Недостаци приступа су сложеност, слабљење поузданости у податке, слаба свежина података, слаба подршка напредних аналитичких метода као што је машинско учење, висока цена решења. Рад предлаже лејкхаус (енг. *lakehouse*) архитектуру као решење, засновану на отвореним форматима за директан приступ подацима, подршком за машинско учење и науку о подацима, и најсавременијим перформансама. Међутим, ова технологија доноси изазове као што су недостатак стручног кадра и зависност од једног пружаоца услуга.

3. ТЕОРИЈСКИ ПОЈМОВИ И ДЕФИНИЦИЈЕ

3.1. Дата-дривен менаџмент

Дата-дривен менаџмент се ослања на доношење одлука заснованих на анализи и интерпретацији података. Компаније са овом стратегијом ослањају се на податке у циљу персонализовања производа и изградње комплетног односа са проспектима и клијентима [7].

3.2. ОЛАП

ОЛАП (енг. *Online Analytical Processing*) системи су класа система за управљање базама података. Оптимизовани су за аналитичке, агрегационе упите са мањим бројем колона над великом количином историјских података, за генерисање аналитичких извештаја и дешборда. Подаци се ређе ажурирају, и не представљају податке у реалном времену. Насупрот томе, ОЛТП системи управљају континуалним током трансакција, константно мењајући податке. Налазе се у многим системима са којима корисници директно интерагују, попут веб апликација [8].

3.3. ЕТЛ и ЕЛТ

Кључни појмови за разумевање ЕТЛ и ЕЛТ процеса су:

- ***Extract*** (Екстракција) - добављање података из апликације и других сервиса
- ***Transform*** (Трансформација) - чишћење, форматирање, агрегирање, обогаћивање, комбиновање и организовање података
- ***Load*** (Учитавање) - Складиштење података на одредиште

ЕТЛ се често користи као појам вишег нивоа апстракције, под који спадају и ЕТЛ и ЕЛТ - учитавање, складиштење и трансформација података. Кључна разлика је што се код ЕТЛ-а трансформација података врши ван складишта података. Код ЕЛТ-а, подаци се трансформишу након учитавања. ЕЛТ постаје стандард како су складишта постала моћнија и јефтинија [9].

3.4. Принципи инкременталног и потпуног освежавања

Приступ потпуног освежавања (енг. *full refresh*) у сваком извршавању трансформација или екстракције поновно процесуира комплетни садржај изворне табеле (или табела), што резултује новом резултујућом табелом. Инкрементални (енг. *incremental*) приступ проналази редове табеле који су креирани или измењени од последње итерације, и обрађује само те редове. Инкрементални приступ, иако сложенији, је најкориснији за обраду великих табела, када је потпуно процесуирање неефикасно [10].

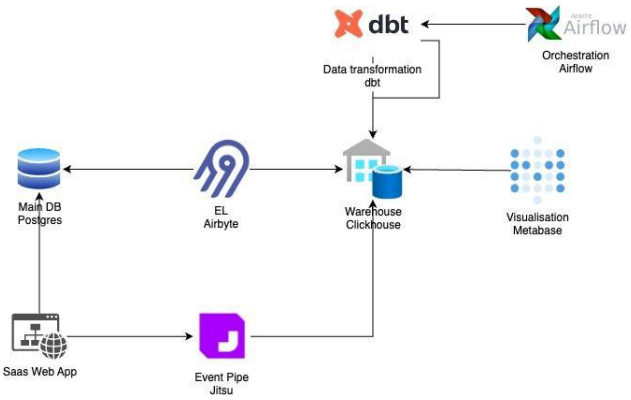
3.5. Аналитике засноване на догађајима

Аналитике засноване на догађајима (енг. *event-based*) служе за праћење и анализирање интеракција између корисника и производа. Догађај је свака акција или понашање које се дешава унутар дигиталне тачке контакта, као што су мобилна апликација, имејл или веб страница. Корисници представљају потенцијалне клијенте који ступају у интеракцију са компанијом у дигиталном окружењу. Када корисници комуницирају са производом, бележи се њихово понашање и кориснички подаци, као што су локација, уређај и

језик. Ова комбинација пружа детаљније информације о томе како и зашто се производ користи.

4. МЕТОДОЛОГИЈА

Дијаграм архитектуре аналитичке платформе представљен је на слици 1.



Слика 1. *Дијаграм архитектуре система*

Архитектура пплатформе подељена је на модуле: ЕЛ и канал за догађаје за прикупљање података, модуле за складиштење, трансформацију, визуелизацију података и модул за оркестрацију. Платформа је организована у модуларну хијерархију *docker compose* конфигурационих фајлова, што омогућава лако управљање и замену модула. Docker compose је алат за дефинисање и извршавање вишеконтејнерских апликација.

4.1. ЕЛ модул

ЕЛ модул као улогу има аутоматизовану екстракцију података са извора и њихово допремање на одредиште. У конкретном случају платформе извор података представља база података софтвер као услуга апликативног система. Централизовано одредиште за податке представља складиште података.

За имплементацију ЕЛ модула коришћен је *Airbyte* [11].

4.2. Канал за догађаје модул

Модул канал за догађаје прати догађаје настале током корисничког коришћења апликације, функционишући као посредник између извора догађаја и одредишта. Улазни догађаји се прикупљају са клијентске или сервер стране, групишу, трансформишу и чувају у систему за складиштење података. За имплементацију модула коришћен је *Jitsu* [12]. Догађаји могу бити подразумевани или специјално дефинисани. У примеру апликације дефинисан је и имплементиран скуп догађаја од интереса за даљу обраду.

4.3. Модул за складиштење података

Модул за складиштење података омогућава централизацију података из различитих извора на једном месту и њихову интеграцију.

За имплементацију изабран је *ClickHouse*. ClickHouse представља добар избор за потребе платформе због своје специфичне архитектуре која пружа оптималне перформансе у ОЛАП сценаријима [13].

4.4. Модул за трансформацију података

Модул за трансформацију података врши интеграцију података из различитих извора у трансформисане податке, погодне за коришћење у аналитичке сврхе. Улаз у модул су подаци различитих извора, а излаз су трансформисани подаци којима крајњи корисници платформе имају приступ.

Модул је имплементиран као независна компонента у архитектури система. Кључна компонента и алат овог модула је *dbt*. dbt је алат који заузима место стандарда у индустрији за трансформацију података [14].

4.5. Модул за оркестрацију

Модул за оркестрацију развијен је за програмирање, заказивање и праћење радних токова (енг. *workflows*). Обавља задатак редовне интеграције података, уз могућност праћења и отклањања грешака.

Као алат за подршку процесу оркестрације изабран је *Apache Airflow* [15], који је скалабилан и омогућава динамичко и прилагодљиво писање радних токова.

4.6. Модул за визуелизацију

Модул за визелизацију представљен је платформом *Metabase* [16] са скупом дешборда и корисничким налозима са одговарајућим правима приступа. Два глава дешборда пружају увид у коришћење апликације: Клијенти и зарада који садржи основне метрике које омогућавају увид у здравље базе претплатника, као и Поглед у резултате кластеровања који представља визуализоване резултате потенцијалне сегментације корисника апликације методом кластеровања. Овај модул је тачка платформе са којом интерагују крајњи корисници платформе - развојни тим, који може укључивати: *product owner*-а, маркетинг тим, тим за продају, вођу тима и друге.

4.7. Интеграција са платформом

Апликације засноване на моделу софтвера као услуге морају се интегрисати са аналитичким платформама. Без обзира на механизам прикупљања података, потребно је прилагодити постојеће апликације кза коришћење њихових података у аналитици, што захтева и сарадњу развојних тимова. За илустрацију процеса интеграције са платформом реализована је пример апликација и представљене неопходне измене које укључују додавање нових зависности апликације, измену њеног изворног кода и прилагођавање базе података ЕЛ алатима.

5. ЕКСПЕРИМЕНТИ

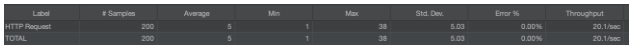
Спроведена је евалуација неколико компоненти платформе: модула за складиштење података са утицајем адекватне имплементације модула за трансформацију на перформансе, као и евалуација модула који представља канал за догађаје. Тестиране су перформансе у просечним и условима високог интензитета коришћења.

У оквиру оба наведена експеримента коришћен је алат *Apache Jmeter* [17]. Резултати тестирања модула за складиштење под вишим оптерећењем од 10 упита од стране 10 паралелних корисника, приказани су на слици 2, изражени у милисекундама и агрегирани за све упите.

Label	# Sampl... ↓	Average	Min	Max	Std. Dev.	Error %	Throughput	Received ...	Sent KB/sec	Avg. Bytes
BI Query ...	56235	169	3	1739	169.33	0.00%	464.7/sec	24.28	0.00	53.5
TOTAL	56235	169	3	1739	169.33	0.00%	464.7/sec	24.28	0.00	53.5

Слика 2. *Резултати тестирања дешборд упита над оптимално моделованом табелом*

Евалуација модула који представља канал за догађаје показује да успешно обрађује захтеве 200 паралелних корисника и допрема догађаје до складишта. Резултати су приказани на слици 3.



Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput
HTTP Request	200	5	1	38	5.03	0.00%	20.1/sec
TOTAL	200	5	1	38	5.03	0.00%	20.1/sec

Слика 3. *Резултати за евалуациону групу од 200 нити*

6. ЗАКЉУЧАК

У раду је представљена модуларна платформа која пружа подршку дата-дривен приступу у развоју софтверских решења, имплементирана у циљу унапређења вредности софтверских решења за крајње кориснике и повећања шансе позитивног исхода приликом пласмана на тржиште.

Евалуација платформе спроведена је над неколико модула платформе, кроз експерименте који су симулирали различите типове оптерећења. Тестирања су потврдила поузданост и ефикасност платформе, доказујући да решење испуњава захтеве за реалну употребу.

Побољшање у односу на постојећу литературу представља коришћење реалних података о корисницима и њиховим навикама. Предложена побољшања укључују подршку напреднијих аналитичких техника, проширење скупа дешборда и метрика које се посматрају. Са инфраструктурног становишта, у зависности од потреба компаније, разликују се и оптималне архитектуре решења, па би се модуларна архитектура решења могла у складу са њима прилагодити или проширити.

7. ЛИТЕРАТУРА

[1] C. Soo, T. Devinney, D. Midgley, и A. Deering, „Knowledge Management: Philosophy, Processes, and Pitfalls“, *Calif. Manage. Rev.*, том 44, изд. 4, стр. 129–150, Јули 2002, doi: 10.2307/41166146.

[2] C.-T. Su, Y.-H. Chen, и D. Y. Sha, „Linking innovative product development with customer knowledge: a data-mining approach“, *Technovation*, том 26, изд. 7, стр. 784–795, Јули 2006, doi: 10.1016/j.technovation.2005.05.005.

[3] L. Bstieler *и остали*, „Emerging Research Themes in Innovation and New Product Development: Insights from the 2017 PDMA-UNH Doctoral Consortium“, *J. Prod. Innov. Manag.*, том 35, изд. 3, стр. 300–307, Maj 2018, doi: 10.1111/jpim.12447.

[4] Y. Zhan, K. H. Tan, и B. Huo, „Bridging customer knowledge to innovative product development: a data mining approach“, *Int. J. Prod. Res.*, том 57, изд. 20, стр. 6335–6350, Окт. 2019, doi: 10.1080/00207543.2019.1566662.

[5] M. Cantamessa, F. Montagna, S. Altavilla, и A. Casagrande-Seretti, „Data-driven design: the new challenges of digitalization on product design and development“, *Des. Sci.*, том 6, стр. e27, 2020, doi: 10.1017/dsj.2020.25.

[6] M. Armbrust, A. Ghodsi, R. Xin, и M. Zaharia, „Lakehouse: A New Generation of Open Platforms that Unify Data Warehousing and Advanced Analytics“, 2021.

[7] SYDLE, „Data-Driven: What It Is and Why It’s Important“, Blog SYDLE. Приступљено: 19. Септембар 2024. [На Интернету]. Available at: https://www.sydle.com/blog/data-driven-what-it-is-and-why-it-s-important-606c8a4e4b136c41e0e2c334

[8] „What Is Online Transactional Processing (OLTP)? | IBM“. Приступљено: 19. Септембар 2024. [На Интернету]. Available at: https://www.ibm.com/topics/oltp

[9] „ETLs, ELTs, and Reverse ETLs“. Приступљено: 19. Септембар 2024. [На Интернету]. Available at: https://www.metabase.com/learn/grow-your-data-skills/analytics/etl-landscape

[10] „Incremental models in-depth | dbt Developer Hub“. Приступљено: 19. Септембар 2024. [На Интернету]. Available at: https://docs.getdbt.com/best-practices/materializations/4-incremental-models

[11] „Airbyte | Open-Source Data Movement for LLMs | AI Platform“. Приступљено: 25. Септембар 2024. [На Интернету]. Available at: https://airbyte.com/

[12] „Jitsu“. Приступљено: 25. Септембар 2024. [На Интернету]. Available at: https://jitsu.com/

[13] ClickHouse, „Fast Open-Source OLAP DBMS“, ClickHouse. Приступљено: 13. Септембар 2024. [На Интернету]. Available at: https://clickhouse.com

[14] „dbt Labs Builds Momentum as the Industry Standard for Data Transformation“, dbt Labs. Приступљено: 15. Септембар 2024. [На Интернету]. Available at: https://www.getdbt.com/blog/dbt-labs-builds-momentum-as-the-industry-standard-for-data-transformation

[15] „Home“, Apache Airflow. Приступљено: 25. Септембар 2024. [На Интернету]. Available at: https://airflow.apache.org/

[16] „Metabase | Business Intelligence, Dashboards, and Data Visualization“. Приступљено: 25. Септембар 2024. [На Интернету]. Available at: https://www.metabase.com

[17] „Apache JMeter - Apache JMeter™“. Приступљено: 17. Септембар 2024. [На Интернету]. Available at: https://jmeter.apache.org/

Кратка биографија:

Тамара Лазаревић рођена је у Шапцу 1997. год. Смер Рачунарство и аутоматика на Факултету техничких наука у Новом Саду уписује 2016. године. Основне студије завршава 2020. године. Исте године потом, на истом факултету уписује мастер студије. контакт: tamaralazarevic@uns.ac.rs

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.9
DOI: https://doi.org/10.24867/31BE23Krsmanovic

PREDIKCIJA TRAJANJA I CENE TAKSI VOŽNJI

TAXI TRIP TRAVEL TIME AND FARE PREDICTION

Natalija Krsmanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – *U radu je predstavljen postupak izrade sistema za analizu i obradu podataka o taksi vožnjama u Njujorku. Korišćena su dva skupa podataka - jedan koji obuhvata podatke o taksi vožnjama i drugi koji sadrži informacije o vremenskim uslovima. Nad ovim skupovima je sprovedeno pretprocesiranje, kako bi se formirao konačan skup podataka za obuku modela. Vršena je predikcija trajanja i cene upotrebom različitih algoritama mašinskog učenja. Sprovedeno je više eksperimenata, a dobijeni rezultati su upoređeni međusobno i sa rezultatima sličnih radova.*

Ključne reči: *analiza i obrada podataka, algoritmi mašinskog učenja, predikcija, taksi vožnje*

Abstract – *The paper presents the process of creating a system for analysing and processing data on taxi rides in New York. Two datasets were utilized – one containing data on taxi rides and the other containing weather data. Preprocessing was performed on these data sets to create the final dataset for model training. Different machine learning algorithms were employed to predict duration and price. Several experiments were conducted, and the results are compared to those from the literature.*

Keywords: *data analysis and processing, machine learning algorithms, prediction, taxi rides*

1. UVOD

Taksi prevoz predstavlja ključni segment urbanog saobraćaja, omogućavajući praktičan i pouzdan način transporta. Cilj ovog rada jeste rešiti problem predikcije trajanja i cene taksi vožnji u Njujorku, sa fokusom na žute taksije u oblasti Menhetn. Precizna predikcija ovih faktora bi omogućila mnogobrojne pogodnosti svim učesnicima u saobraćaju. Vozači bi imali mogućnost da izaberu najefikasnije rute i smanje vreme čekanja, putnici bi imali realističnija očekivanja na osnovu transparentnosti u cenama i vremenu, a taksi kompanijama bi bila zagarantovana bolja iskorišćenost resursa i zarada. U radu su izabrani sledeći modeli: linearna regresija, model nasumične šume (engl. *Random Forest*), *XGBoost* i višeslojni perceptron (engl. *Multi-Layer Perceptron - MLP*). Po uzoru na prethodna rešenja i slične radove, za mere evaluacije posmatrane su koren srednje kvadratne greške (engl. *Root Mean Square Error - RMSE*) i koeficijent determinacije (engl. *R-squared* - R^2). Najbolje rezultate sa *RMSE* od 2.45 za cenu i 3.98 za trajanje vožnje je imao *XGBoost* algoritam.

U narednom poglavlju su predstavljeni radovi koji se bave rešavanjem sličnog problema, a koji su u najvećoj meri poslužili kao inspiracija za ovaj rad. Treće poglavlje sadrži opis metodologije i upotrebljenih modela. Naredno poglavlje broj 4 predstavlja analizu dobijenih rezultata, dok poslednje, 5. poglavlje iznosi zaključak rada.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Jelena Slivka, vanr.prof.

2. PREGLED STANJA U OBLASTI

Predikcija trajanja vožnje i cene taksi usluga postaje sve relevantnija tema istraživanja usled urbanizacije gradova i dinamičkih promena u saobraćaju. Shodno tome, poslednjih godina veliki broj radova se fokusira na rešavanje ovog problema predlažući različite metode i tehnike. U ranim fazama istraživanja, predikcija je bila zasnovana na klasičnim statističkim modelima, kao što su linearni regresioni modeli i modeli vremenskih serija. Vremenom postaju popularni modeli bazirani na stablima, potom neuronske mreže i u prethodnim godinama modeli dubokog učenja i hibridni modeli.

Zadatak autora u radu [1] jeste predikcija cene i trajanja taksi vožnje u Njujorku korišćenjem podataka iz skupa podataka *„New York City Taxi and Limousine Commission's trip data"*. Predloženi modeli su linearna regresija i model nasumične šume. Nakon izvršene analize uticaja obeležja, došli su do zaključka da su prosečna brzina, broj vožnji u satu, lokacija odredišta i dužina pređenog puta, obeležja od najvećeg značaja za predikcije modela. Ovi rezultati usmerili su fokus eksplorativne analize u ovom radu na pomenuta obeležja. Posmatrajući koren srednje kvadratne greške kao meru evaluacije, model nasumične šume je pokazao bolje rezultate u odnosu na linearnu regresiju i oni iznose 2.28 za cenu i 5.24 za trajanje.

Par godina kasnije, nad istim skupom podataka, vršena je takođe predikcija trajanja vožnje u radu [2]. Pored osnovnog skupa podataka sa taksi vožnjama, korišćen je skup podataka o vremenskim uslovima preuzet sa veb sajta *„National Wheater Service*". Izabrani su modeli bazirani na stablima, i to: *CART* (*engl. Classification And Regression Tree*), *Random Forest*, *Extra Trees*, *XGBoost* i *LightGBM*. Ono što ovaj rad dodatno uvodi jeste kratkoročna (*short-term)* predikcija, koja u obzir uzima podatke od poslednjih *i* vožnji, gde je uočeno da na trajanje vožnje najviše utiču informacije iz prethodnog sata. Rezultati za dugoročnu (*long-term)* predikciju pokazuju da najmanja greška iznosi 4.22 za *XGBoost*

model, dok se najlošijim pokazao *CART*. Iz tog razloga, jedan od korišćenih modela u ovom radu je *XGBoost*.

S obzirom na razvoj metoda mašinskog učenja, autori rada [3] pored stabala uvode i neuronske mreže za predikciju trajanja taksi vožnje. Konkretno koriste *XGBoost* i *MLP*. Ovaj rad se izdvaja po uvođenju novih obeležja, a to su *Manhattan*, *Haversin* i *Bearing* rastojanja, izračunata uz pomoć početnih i krajnjih koordinata. Dodatno je primenjen *K-means* algoritam radi podele vožnji u klastere u kojima je transport izvršen, što je rezultovalo sa dodatnih 200 obeležja i dodatnim povećanjem preciznosti modela. Tačnost modela merena je sa *RMSE*, koja iznosi 0.41 za *XGBoost*, a 0.44 za *MLP*.

3. METODOLOGIJA

U ovom poglavlju biće predstavljen proces implementacije sistema za predviđanje trajanja i cene taksi vožnji. S obzirom da su trajanje i cena numerička obeležja sa kontinualnim vrednostima, ovaj problem se definiše kao regresioni problem. U nastavku će biti opisani upotrebljeni skupovi podataka, način na koji je vršena analiza i potom način kreiranja i obučavanja modela.

3.1. Skupovi podataka

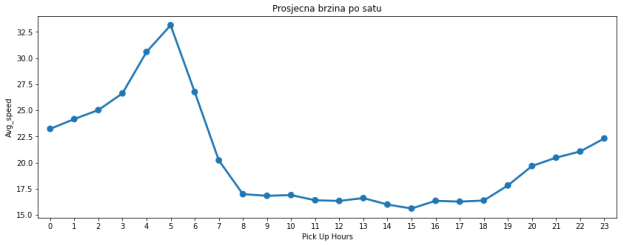
Po uzoru na pomenute radove iz literature, za glavni skup podataka izabran je „*NYC TLC (New York City Taxi and Limousine Commission) Trip Record Data*" [4]. Ovaj skup sadrži podatke o vožnjama žutih i zelenih taksija u Njujorku od 2009. godine do danas. Uključuje informacije o datumu i vremenu početka i kraja vožnje, broju putnika, ukupnom pređenom putu, početnoj i krajnjoj lokaciji, ceni, taksama i drugim obeležjima vezanih za dodatne naplate. Za potrebe ovog rada izabrani su podaci za žute taksije iz maja 2016. godine, koji sadrže 11 832 050 uzoraka i 19 obeležja. Na osnovu rezultata iz rada [2] primećeno je da se predikcija poboljšava uvođenjem dodatnih informacija o vremenskim uslovima. Kao rezultat toga, u analizu je uključen i skup podataka „*Historical Hourly Weather Data*" [5], koji sadrži podatke o temperaturi, vazdušnom pritisku, vlažnosti vazduha, brzini vetra i kratak opis vremena, za 30 američkih gradova, u periodu od 2012. do 2017. godine.

3.2. Pretprocesiranje i analiza podataka

Nad prethodno pomenutim skupovima podataka je vršen proces obrade i analize podataka, sa ciljem poboljšanja kvaliteta podataka i rezultata predikcije.

U prvom skupu podataka izbačene su nedostajuće vrednosti i nerelevantni podaci kao što su takse, putarine i drugi fiksni troškovi. Obeležja koja su sadržala datum i vreme su razdvojena u tri nova: datum, vreme i sat. Pored toga, dodato je novo obeležje koje ukazuje na to da li je u pitanju radni dan, vikend ili praznik. Takođe je dodata informacija o vremenskom periodu putovanja. Ciljno obeležje, trajanje vožnje, nije bilo direktno dostupno ali je izračunato uz pomoć početnog i krajnjeg vremenskog trenutka.

Nakon analize uočeno je da postoje vožnje kod kojih je dužina pređenog puta nula, a trajanje i cena negativne vrednosti. Ti uzorci su izbačeni, kako ne bi remetili obučavanje modela i predikciju. Prilikom posmatranja odnosa trajanja vožnje i pređenog puta, uočeno je da postoje uzorci kod kojih je pređeni put blizu nula kilometara, a vremena trajanja se kreću i preko 10 sati. Ti uzorci imaju nerealno male vrednosti srednje brzine, pa su zbog toga eliminisani oni sa prosečnom brzinom manjom od 5 km/h. Nakon ove transformacije odnos trajanja i distance postaje jasno definisan, pri čemu se uočava očekivana pozitivna korelacija između njih. Na slici 1 prikazana je raspodela prosečne brzine po satima tokom dana. Može se primetiti da su najveće vrednosti u noćnim časovima, što je i očekivano zbog manjeg intenziteta saobraćaja. Nasuprot tome, skoro dvostruko niže vrednosti za brzinu su između 08:00 h i 19:00 h. Iz ovoga se može zaključiti da gužve u saobraćaju imaju značajniji uticaj na tok vožnji i brzinu kretanja, u odnosu na ograničenja brzine. Pored toga, potvrđena je pretpostavka da su prosečne brzine kretanja znatno veće tokom vikenda naspram onih tokom radnih dana.



Slika 1. *Raspodela prosečne brzine po satima*

S obzirom na to da su podaci o vremenskim uslovima iz drugog skupa raspoređeni po odvojenim fajlovima, prvobitno je izvršeno spajanje podataka za maj 2016. godine. Nakon toga su izdvojena obeležja za datum i vreme, na osnovu kojih je proširen prvi skup podataka. U najvećem broju zabeleženih vožnji, vremenski uslovi su bili magloviti, oblačni ili sa blagim padavinama. Iako se po rezultatima iz sličnih radova očekuje da vremenske karakteristike utiču na ciljna obeležja, na osnovu matrice korelacije za sva obeležja proširenog skupa podataka, primećuje se veoma slaba korelacija pomenutih.

3.3. Priprema podataka za modele

Za upotrebu određenih modela za vršenje predikcije potrebno je obezbediti da sva obeležja budu numeričkog tipa. Međutim, u drugom skupu podataka postoji obeležje koje ukratko opisuje vremenske uslove za dati trenutak. Za njega je dostupno 15 različitih opisa odnosno kategorija. Slične kategorije su grupisane zajedno, što je dovelo do konačnih pet kategorija. Za njihovo konvertovanje u numeričke vrednosti iskorišćen je *One Hot Encoding* [6]. Rezultat ovog koraka jeste novih pet obeležja, odnosno skup podataka sa 21 obeležjem. Dati skup se za potrebe obučavanja i testiranja modela, prvobitno deli na trening i test skup u razmeri 80:20. Potom se isti metod primenjuje na dobijeni trening skup, gde se 10% podataka izdvaja u validacioni skup, a preostali deo označava konačan trening skup.

Poslednji korak pre obučavanja modela jeste standardizacija obeležja. Ovim postupkom se vrši skaliranje na takav način da svako obeležje ima srednju vrednost 0, a standardnu devijaciju 1. Na ovaj način je završen proces pripreme podataka.

3.4. Prediktivni modeli

U uvodnom delu ovog rada nabrojani su algoritmi mašinskog učenja koji će biti korišćeni za predviđanje cene i trajanja taksi vožnji. Pored izbora adekvatnih algoritama za konkretan problem, potrebno je konstruisati modele na način da vrše što preciznije predikcije. Ovaj proces uključuje treniranje modela, odnosno optimizaciju hiperparametara koji utiču na njegovo ponašanje i odluke.

Hiperparametri su optimizovani na validacionom skupu za svaki model posebno. Prva mera evaluacije jeste koren srednje kvadratne greške (engl. *Root Mean Squared Error- RMSE),* koja vrednosti prikazuje u jedinicama obeležja za koji se posmatra. Iako *RMSE* naglašava velike greške i otkriva autlajere, ne daje informaciju o tome koliko dobro model objašnjava ukupnu varijabilnost u podacima. Zbog toga se kao druga mera evaluacije uvodi koeficijent determinacije.

Za lineranu regresiju posmatrane su tri hipoteze. Prva predstavlja osnovni oblik linerane regresije, druga sadrži interakcije između nezavisnih obeležja, a treća pored interakcija sadrži i polinomijalne članove drugog reda. Kao što i očekivano od navedenih najbolje rezultate ima polinomijalna regresija reda 2.

Sledeći model koji je korišćen jeste model nasumične šume, za koga je vršena optimizacija broja stabala i dubine stabla. Prateći vrednosti *RMSE*, najveći pad se dešava za dubinu 15. Zbog toga su konačne vrednosti hiperparametara postavljene na 15 za dubinu stabla i 30 za broj stabala.

S obzirom na ograničene računarske resurse i veliki broj hiperparametara koji se podešavaju za *XGBoost*, vrednosti za njih nisu kombinovane i posmatrane zajedno. Prvobitno je posmatran odnos dubine stable (*max_depth*) i broja uzoraka po listu (*min_child_weights)*, jer oni najviše utiču na arhitekturu modela i oblik stabala. Njih je potrebno zajedno određivati kako bi se uspostavila balansiranost između varijanse i pristrasnosti modela. Nakon njih su podešavani procenat uzoraka (*subsample)* i procenat obeležja (*colsample_bytree)* koji se koriste pri izgradnji svakog stabla i na samom kraju brzina učenja (*eta)*. Konačne vrednosti za njih iznose: *max_depth* = 9, *min_child_weights* = 6, *subsample* = 1, *colsample_bytree* = 0.8 i *eta* = 0.2.

Na osnovu rezultata prethodnih modela, za *MLP* isprobane su različite kombinacije broja skrivenih slojeva i aktivacione funkcije. Zanimljivo je da je za svaku strukturu skrivenih slojeva aktivaciona funkcija *tanh* dala bolje rezultate. Konačna arhitektura jeste (20, 20, 20) – tri skrivena sloja od po 20 neurona.

4. REZULTATI I DISKUSIJA

U ovom poglavlju biće predstavljeni rezultati eksperimenata sprovedenih nad različitim skupovima podataka, primenjujući izabrane modele.

Tabela 1 prikazuje vrednosti *RMSE* svih modela za cenu taksi usluge, nad tri različita skupa podataka. Prvi predstavlja inicijalni skup podataka o taksi vožnjama koji je pročišćen od nedostajućih vrednosti i standardizovan. Naredni skup je isti kao prethodni uz dodatno transformisanje i obrađivanje uz pomoć određenih tehnika. Treći skup je nastao spajanjem prethodnog skupa sa vremenskim uslovima, kako bi se istražilo koliko dodatne informacije o vremenu utiču na tačnost i unapređenje modela. Kada se porede mere evaluacije među skupovima, primetno je da je *RMSE* za sve modele najveća i mnogo odskače za inicijalni nepročišćeni skup. Ovi rezultati ukazuju na to da modeli nisu uspeli adekvatno da uče iz podataka zbog prisustva šuma i autlajera koji su narušavali strukturu podataka u prvom slučaju. Najbolji rezultati za sve modele se vide u poslednjem redu, koji se odnosi na prošireni skup podataka.

Tabela 1. Prikaz *RMSE* za sve modele obučene i testirane nad tri različita skupa podataka za cenu taksi vožnje

Skupovi podataka	**Model**			
	Linearna regresija	**Model nasumične šume**	***XGBoost***	***MLP***
Skup 1	12.198	6.51	6.548	7.039
Skup 2	3.204	2.522	2.478	2.704
Skup 3	3.198	2.519	**2.458**	2.665

Rezultati pokazuju da iako *MLP* model predstavlja savremeniji pristup i kasnije je počeo da se koristi za rešavanje ovakvih problema, njegove performanse su slabije u poređenju sa algoritmima zasnovanim na stablima učenja. Pretpostavlja se da model nasumične šume i *XGBoost* prave manje greške jer bolje upravljaju interakcijama između obeležja, zbog načina na koji dele podatke i kreiraju stabla, dok MLP uči interakcije na globalnom nivou kroz optimizaciju težina. U predikciji trajanje taksi vožnje primetan je isti trend u vrednostima mera evaluacije kao i kod cene vožnje. Takođe i u ovom slučaju najmanju *RMSE* ima *XGBoost* algoritam sa vrednošću 3.98. Na osnovu pomenutih rezultata uočava se da modeli generalno vrše bolje predikcije za cenu u odnosu na trajanje vožnje. To se može objasniti činjenicom da je cena više korelirana sa distancom, koja predstavlja najvažnije obeležje prilikom obučavanja modela.

Pored međusobnog poređenja modela, izvršeno je poređenje najuspešnijeg modela u ovom radu sa odgovarajućim modelima iz sličnih radova. Ovakvo poređenje omogućava da se sagleda da li pristup iz ovog rada nadmašuje već postojeća rešenja i kako može dodatno da se unapredi.

Za model linearne regresije, vrednosti *RMSE* u ovom radu su veće za oba ciljna obeležja u odnosu na rad [1]. Iako postoji razlika, ona je jako mala, odnosno modeli predviđaju na sličan način. Ovakvi rezultati su i očekivani jer su u oba rada korišćeni isti podaci o taksi vožnjama za isti vremenski period.

Rezultati u tabeli 2 pokazuju da je model nasumične šume, prikazan u ovom radu, približan najboljima za oba izlazna obeležja. Zanimljivo je poređenje sa zadnjim radom jer je u njemu model nasumične šume podešen tako da koristi sve hiperparametre sa njihovim podrazumevanim vrednostima. Pošto je razlika u *RMSE* čak 1.5, sledi zaključak da je podešavanje vrednost

određenih hiperparametara uticalo na poboljšanje u odnosu na podrazumevane.

Tabela 2. Poređenje *RMSE* rezultata iz ovog rada sa sličnim rešenjima koristeći model nasumične šune

Model nasumične šume		
Model	**Cena ($)**	**Trajanje (min)**
Model prikazan u ovom radu	2.519	4.851
Model u radu [1]	**2.287**	5.240
Model u radu [2]	-	**4.232**
Model u radu [7]	3.953	-

Što se tiče rezultata za *XGBoost*, najbolji rezultati i veća razlika se zapaža u radu [3] gde *RMSE* iznosi samo 0.41 za predikciju trajanja. Pretpostavka je da je ovo posledica činjenice da se u tom radu koriste podaci u periodu od tri godine, pri čemu je izvršena i klasterizacija podataka. Takođe dodata su nova obeležja koja predstavljaju različite načine izračunavanja distance, pa je samim tim model bolje uspeo da se prilagodi podacima i izvrši približniju predikciju.

Na kraju su upoređeni rezultati *MLP* modela. Bitno je istaći da su arhitekture modela u ovom radu i radu [8] identične- mreža sadrži tri skrivena sloja sa po 20 neurona. Uprkos tome, *MLP* iz ovog rada je ostvario nešto bolje performanse, što može biti posledica različitih tehnika pretprocesiranja podataka.

5. ZAKLJUČAK

U ovom radu je rešavan problem predviđanja cene i trajanja taksi vožnje, sa fokusom na Njujork, grad poznat po velikoj upotrebi taksi usluga. Osnovni cilj sistema je pružanje tačnijih predikcija za ovu vrstu usluge, što bi značajno poboljšalo organizaciju svakodnevnih aktivnosti čoveka i omogućilo bolju optimizaciju vremena u okruženju urbanog života. Implementacija ovog sistema podrazumeva kombinovanje više modula sa definisanim ulogama. Započeto je sa prikupljanjem podataka, gde je prvi skup sadržao informacije o vožnjama za žute taksije u toku maja meseca 2016. godine. Drugi skup podataka je obuhvatao vremenske uslove za isti period, prikupljanje na svakih sat vremena. Isprobani su linearna regresija, model nasumične šume, *XGBoost* i *MLP*. Za svaki od modela izvršena je optimizacija hiperparametara Za potrebe treniranja ali i testiranja modela, skup podataka je podeljen na trening, validacioni i test skup. Budući da je u pitanju regresioni problem, za mere evaluacije su izabrane koren srednje kvadratne greške i koeficijent determinacije.

Poredeći performanse modela za više različitih kombinacija ulaznih obeležja i podataka, svi su se najviše obučili na proširenom skupu podataka. Najbolji rezultati za oba ciljna obeležja su postignuti upotrebom *XGBoost* algoritma, čija *RMSE* za trajanje iznosi 3.48, a za cenu vožnje 2.46. Analizom rezultata dolazi se do zaključka da su svi modeli osetljivi na anomalije u podacima, odnosno da ne uspevaju da nauče odnose između podataka i obeležja ukoliko se nad inicijalnim podacima ne odradi pretprocesiranje. Dodatno je uočeno da dodavanje vremenskih karakteristika nije doprinelo performansama modela u onoj meri koja se očekivala čitajući druge radove iz literature.

Na osnovu rezultata iz rada [3], jedan od načina za unapređenje ovog sistema jeste proširivanje skupa podataka. To znači da bi bilo potrebno posmatrati vožnje za više meseci ili čak za više godina ukoliko to računarski resursi dozvoljavaju. Pored navedenog, sistem bi se mogao poboljšati dodavanjem više različitih izvora podataka, poput podataka o gustini saobraćaja, radovima na putevima, događajima u gradu koji mogu uticati na saobraćajnu gužvu i drugim za određeni vremenski trenutak. Takođe dodatno proširenje bi moglo obuhvatiti bolji pregled geografskih podataka time što bi sistem uzeo u obzir različite karakteristike različitih delova grada.

6. LITERATURA

[1] C. Antoniades, Delara Fadavi, Antoine Foba Amon, "*Fare and Duration Prediction: A Study of New York City Taxi Ride",* Semantic Scholar, 43844792, 2016.

[2] Huang, H., Pouls, M., Meyer, A., Pauly, M, " *Travel Time Prediction Using Tree-Based Ensembles*", Lecture Notes in Computer Science, vol 12433, pp 412–427, 2020.

[3] Poongodi M, Malviya M, Kumar C, "*New York City taxi trip duration prediction using MLP and XGBoost"*, International Journal of System Assurance Engineering and Management, 13(Suppl 1), 16-27, 2022.

[4] https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page (pristupljeno u avgustu 2024.)

[5] https://www.kaggle.com/datasets/selfishgene/historical-hourly-weather-data/code (pristupljeno u avgustu 2024.)

[6] https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/ (pristupljeno u avgustu 2024.)

[7] *https://github.com/jahnavi-chowdary/New-York-Taxi-Fare-Prediction/tree/master* (pristupljeno u septembru 2024.)

[8] https://github.com/raymonduchen/MLND-P6-New-York-City-Taxi-Fare-Prection/tree/master (pristupljeno u septembru 2024.)

Kratka biografija:

Natalija Krsmanović rođena je 1999.godine u Gradišci, BiH. Osnovne akademske studije završila je 2022. godine na Fakultetu tehničkih nauka, na kom brani master rad 2024. godine iz oblasti Računarstvo i automatika – Inteligentni sistemi.
kontakt:
krsmanovic.natalija99@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.38
DOI: https://doi.org/10.24867/31BE25Petrovic

SISTEM ZA DETEKCIJU GRUBE VOŽNJE BAZIRAN NA TROSLOJNOJ ARHITEKTURI

DRIVING STYLE DETECTION SYSTEM BASED ON THREE-TIER ARCHITECTURE

Aleksandar Petrović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom članku opisana je realizacija sistema za detekciju grube vožnje. Pod grubom vožnjom se smatra naglo kočenje, naglo ubrzanje i grubo skretanje. Sam sistem se sastoji iz tri dela: aplikacije implementirane na ESP32 mikrokontroleru, Android aplikacije i veb servera. Esp32 deo služi za pribavljanje OBD2 podataka iz vozila. Android aplikacija beleži događaje koje smatramo grubom vožnjom zajedno sa koordinatama na kojima su se desili. Da bi sistem za detekciju grube vožnje uspešno funkcionisao potrebno je podatke sa ESP32 i telefona prikupiti i učiniti dostupnim korisniku u tu svrhu razvijen je veb server.*

Ključne reči: *Android, ESP32, CAN, OBD2, MySQL, JavaScript, Node.js, HTML, CSS, GPS, TWAI*

Abstract – *This article describes the implementation of driving style detection system. Events that are interesting for us are hard braking, sudden acceleration and sharp turning. This project consists of three parts: application implemented on ESP32 microcontroller, Android app and web server. Part implemented on ESP32 is responsible for collecting vehicle-related data. The Android app is used for detecting events that are considered important for evaluating driving style. For system to function properly data from both the ESP32 and the phone must be gathered and made accessible to the user, for which the web server was developed.*

Keywords: *Android, ESP32, CAN, OBD2, MySQL, JavaScript, Node.js, HTML, CSS, GPS, TWAI*

1. UVOD

Transport ljudi i robe je neizostavni deo funkcionisanja savremenog društva. Problem porasta nezgoda u saobraćaju dovodi do velikih finansijskih gubitaka kao i do lakših i težih povreda učesnika u saobraćaju. Nasilnička i gruba vožnja jedan je od glavnih razloga nezgoda.

Posmatrano iz ugla vlasnika kompanija sa velikim brojem vozila, informacije o stilu vožnje zaposlenih omogućavaju preventivno reagovanje i uštedu novca.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Predrag Teodorović, vanredni profesor.

Da bi gruba vožnja mogla da se detektuje prvo je moramo definisati. Pod grubom vožnjom se smatra naglo kočenje, naglo ubrzanje i grubo skretanje, stvari koje se lako mogu detektovati uz pomoć akcelerometra, dok bi uz to korisno bilo znati i lokaciju gde se sam prekršaj desio, to su podaci koji se lako mogu dobiti sa senzora pametnih telefona. Pored tih podataka korisni podaci bi bili i oni koje vozilo daje tokom same vožnje.

Projektovani sistem se sastoji iz tri dela:

1. *Android* aplikacije – *GPS* senzor i akcelerometar u samom telefonu su pogodni za detekciju, pošto se sama pozicija telefona u automobilu ne zna pre nego što praćenje krene neophodno je koordinatni sistem akcelerometra rotirati tako da se poklapa sa koordinatnim sistemom vozila
2. Aplikacije na *ESP32* mikrokontroleru – *TWAI/CAN* kontroler unutar same pločice uz dodatni transiver se može koristiti za čitanje *OBD2* podataka iz samog automobila (broj šasije, obrtaji motara, brzina vozila,...), ti podaci sa sobom nose dodatne informacije o samoj vožnji
3. Veb servera – koji služi za vizuelizaciju podataka dobijenih od ostatka sistema, gde se mogu videti sve rute koje su vožene, zajedno sa svim prekršajima koji su se desili na tim rutama, takođe se mogu videti poslednji pristigli podaci koje je poslao ESP32 mikrokontroler za izabrano vozilo i gde se može podešavati interval semplovanja podataka koje vrši ESP32

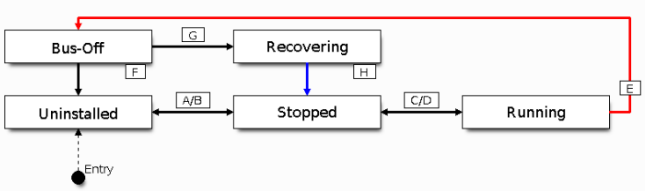
2. *ESP32*

Ovo poglavlje bavi se komunikacijom sa vozilom, protokolima koji se u tu svrhu koriste kao i načinom na koji je željena funkcionalnost implementirana.

2.1. *TWAI*

Deo sistema koji komunicira sa vozilom realizovan je korišćenjem *ESP32* mikrokontrolera. *ESP32* mikrokontroleri su pogodni za aplikacije u kojima se koristi *CAN* interfejs jer u sebi imaju integrisan *TWAI* (eng. *Two-Wire Automotive Interface*) kontroler koji je kompatibilan sa *ISO11898-1* protokolom i *CAN2.0* interfejsom. *TWAI* je serijski protokol koji je asinhron, ima mogućnost detekcije grešaka i obaveštavanje ostalih nodova o tome, podržava više mastera, tako da transfer

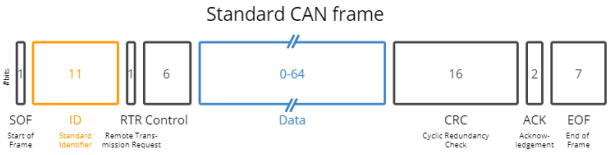
može biti iniciran od strane bilo kog noda, a svi ostali nodovi će dobiti poruku [1]. Da bi se *ESP32* koristio na *TWAI/CAN* magistrali potrebno mu je dodati eksterni transiver kao što je *MCP2551*. *TWAI* kontroler poseduje 4 signala *TX, RX, BUS-OFF i CLKOUT*, od koji su samo prva dva obavezna. Nizak logički nivo (0V) smatra se dominantim i na prijemnom i na predajnom pinu. Svaka *TWAI* poruka sastoji se od sledećih elemenata: identifikatora, dužine korisne poruke - *DLC (eng. Data Length Code)* i 8 bajtova podataka korisne poruke. Rad drajvera kontrolera je prikazan na slici 1.



Slika 1. *Prikaz stanja TWAI kontrolera [1]*

2.2. *CAN*

CAN (eng. Controller Area Network) je jedan od najčešće korišćenih standarda za komunikaciju između različitih elektronskih komponenti u automobilu (eng . *Electronic Control Units - ECU*). Nastao je osamdesetih godina prošlog veka, da bi se eliminisale mane direktne, analogne komunikacije između nodova. *CAN* magistrala predstavlja komunikacioni sistem baziran na porukama, koji omogućava brzu i pouzdanu interakciju između različitih delova automobila. Svaki modul može da prima poruke, ali i da ih šalje svim ostalim modulima u mreži (eng. *broadcast-based* protokol) [2]. Svaki *ECU* za sebe odlučuje da li mu je neka poruka potrebna ili ne. Arbitraža se vrši dodelom prioriteta porukama, manji identifikacioni broj poruke, znači da je njen prioritet veći. Svaki podatak koji se šalje na magistrali nalazi se u frejmu, koji sadrži jasno definisana polja, prikazana na slici 2.



Slika 2. *Izgled jednog CAN frejma [2]*

U zavisnosti od performansi postoji nekoliko vrsta *CAN* protokola, a u ovom radu se koristi *High-speed CAN 2.0* koji podržava brzinu prenosa do 1 *Mbit/s* (*boud rate*) i fiksnu veličinu podataka do 8 bajtova. *CAN* je u skladu sa *ISO11898* podeljen u dve podgrupe: *ISO11898-1* za opis sloja za prenos podataka (eng. *data link layer*) i *ISO11898-2* za opis zahteva fizičkog sloja (eng. *physical layer*).

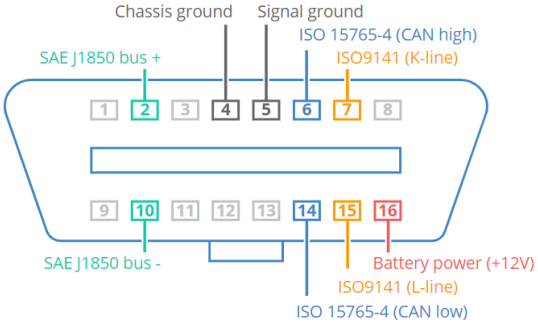
CAN fizički gledano čini par uvrnutih žica, *CAN High* i *CAN Low*. *CAN High* se obično nalazi u opsegu od 2.5V do 3.75V, dok je *CAN Low* od 1.25V do 2.5V. Ukoliko obe linije očitavaju 2.5V signal je recesivan (nivo logičke jedinice), dok ukoliko očitavaju različite vrednosti smatra se da je signal dominantan (logička nula). Važnost prenosa putem ovakvog diferencijalnog signala leži u otpornosti na šum, a u cilju očuvanja integriteta poruke.

2.3. *OBD2*

OBD2 (eng. *On Board Diagnostic*) je standardizovani protokol koji služi za iščitavanje kodova grešaka (eng. *DTC - Diagnostic Trouble Codes*) i podataka vezanih za trenutno stanje automobila (eng. *current data*) preko *OBD2* konektora. [3] Odnos *OBD2* i *CAN* je takav da je *OBD2* protokol višeg nivoa, a *CAN* predstavlja metod komunikacije. *OBD2* definiše da:

- Brzina na *CAN* magistrali mora biti 250K ili 500K bita po sekundi
- Identifikatori poruka moraju imati jedanaest (u automobilima) ili dvadeset i devet bita i koriste se u komunikaciji
- Poruka mora biti dužine do osam bajtova

Postoje dva tipa *OBD2* konektora, tip A i tip B, najveća razlika je napon na 16. pinu koji kod tipa A iznosi 12V, a kod tipa B 24V. Na slici 3. Prikazan je konektor tipa A, koji se sastoji od 16 pinova, gde su pinovi 6 i 14 određeni za *CAN High* i *CAN Low*.



Slika 3. *OBD2 konektor tipa A [3]*

OBD2 PID (eng. *Parameter IDs*) su predefinisani kodovi koji se koriste za traženje podataka od automobila zajedno sa servisima, to jest modovima. *OBD2* zahtevi kao što su zahtevi za broj šasije (eng. *Vehicle Identification Number - VIN*) i zahtevi za iščitavanje kodova grešake u svom odgovoru imaju više od osam bajtova, a kako jedna poruka u protokolu može imati maksimalno osam bajtova, potrebno je koristiti takozvani *ISOTP* (*ISO 15765-2*) transportni protokol. *ISOTP* definiše četiri različite vrste frejma: *Single frame*, *First frame*, *Consecutive frame* i *Flow control frame*.

2.4. Implementirani softver

Implementirani softver podeljen je na dva dela:

- *HAL* (eng. *Hardware Abstraction Layer*) – u kom su implementirani delovi softvera zaduženi direktno za hardver
- *MainApp* – u kom je implementirana glavna funkcionalnost

HAL u sebi sadrži četiri modula: modul koji je zadužen za funkcije koje se koriste za konekciju na internet, modul za logovanje, modul zadužen za sinhronizaciju vremena sa serverom (*„pool.ntp.org“*) i modul zadužen za *CAN* funkcije. Softver implementiran u ovom sloju radi direktno sa *API*-jem (eng. *Application programming interface*) koji je *ESP-IDF* pružio i vrši svu potrebnu inicijalizaciju i proveru podataka kako se na višem sloju ne bi vodilo računa o tome.

MainApp je implementirana kroz dve mašine stanja, jednu koja kontroliše pristup internetu i mašinu stanja koja implementira komunikaciju sa automobilom.

3. ANDROID

Deo sistema koji prikuplja podatke sa akcelerometra i *GPS*-a implementiran je u okviru *Android* aplikacije. *Android* je prvenstveno platforma za mobilne telefone koja se koristi za različite namenske uređaje. Za razvoj *Android* aplikacije korišćen je *Java* programski jezik i *Android Studio* kao razvojno okruženje.

3.1. Senzori

Senzori u mobilnim telefonima su malih dimenzija i potrošnje i spadaju u grupu mikro-elektro-mehaničkih senzora (eng. *micro-electro-mechanical sensors – MEMS*). U ovom radu je korišćen akcelerometar i *GPS* prijemnik.

Akcelerometar

Akcelerometar je sastavni deo svakog mobilnog telefona i predstavlja uređaj za merenje ubrzanja (akceleracije). Meri promene sila koje utiču na uređaj u metrima po sekundi na kvadrat u tri ose (*x, y* i *z*).

GPS

GPS (eng. *Global Positioning System*) je globalni navigacioni satelitski sistem. *GPS* prijemnik (u mobilnom telefonu) može na osnovu radio signala koje prima sa *GPS* satelita da odredi svoju tačnu lokaciju, zahvaljujući činjenici da je brzina kretanja radio talasa kroz vazduh poznata i konstantna. Naravno, da bi proračun bio moguć, potrebni su podaci od minimalno tri satelita.

3.2. Implementirana *Android* aplikacija

Na slici 4. dat je prikaz početne aktivnosti aplikacije. Kako bi prešao na glavnu aktivnost korisnik mora da unese *VIN* u očekivanom formatu. Jedan od primera formata koje se očekuje je 2T3-RFREV7D-W108177.

Slika 4. *Početna aktivnost aplikacije*

Na slici 5. Dat je izgled glavnog dela aplikacije u poljima ***Lat***, ***Lon***, ***Altitude***, ***Speed*** i ***Address*** upisani su podaci dobijeni sa lokacionih senzora.

Pritiskom na dugme ***START TRIP*** počinje prikupljanje podataka sa senzora lokacije i beleže se koordinate početne lokacije. Pritiskom na ***STOP TRIP*** zaustavljamo praćenje vožnje i šaljemo podatke na server, sa svim zabeleženim događajima, tačnije prekršajima naglog kočenja, ubrzanja ili skretanja i koordinate početka vožnje i krajnje destinacije.

SHOW MAP dugme nam omogućava da vidimo mapu svih zabeleženih događaja, dok nam ***SHOW OBDII DATA*** pribavlja podatke koji su poslednji pristigli sa vozila čiji smo *VIN* uneli u početnoj aktivnosti.

Slika 5. *Izgled glavne aktivnosti*

Takođe u samoj aplikacije je moguća i vizuelizacija podataka sa akcelerometra preko grafa i direktnih vrednosti za ose koje će biti ispisane na ekranu.

4. VEB SERVER

U ovom poglavlju biće reči o realizaciji veb servera. Da bi sistem za detekciju grube vožnje uspešno funkcionisao, potrebno je podatke sa senzora mikrokontrolera i pametnog telefona prikupiti i ceo sistem povezati u jedinstvenu celinu. Veb server je podeljen na dve celine, serversku stranu (*back-end*) i klijentsku stranu (*front-end*).

4.1. *Back-end*

Back-end ili serverski deo veb sajta može se podeliti na dve celine veb server i bazu podataka. Uloga mu je da odgovara na *HTTP* (eng. *HyperText Transfer Protocol*) zahteve korisnika. Klijent i server komuniciraju po principu: klijent šalje zahtev, server ga prihvata i na njega odgovara. *Back-end* je realizovan koristeći funkcije koje obezbeđuje *Express.js (Node.js framework)*. [4] Za skladištenje informacija korišćena je *MySQL* baza podataka.

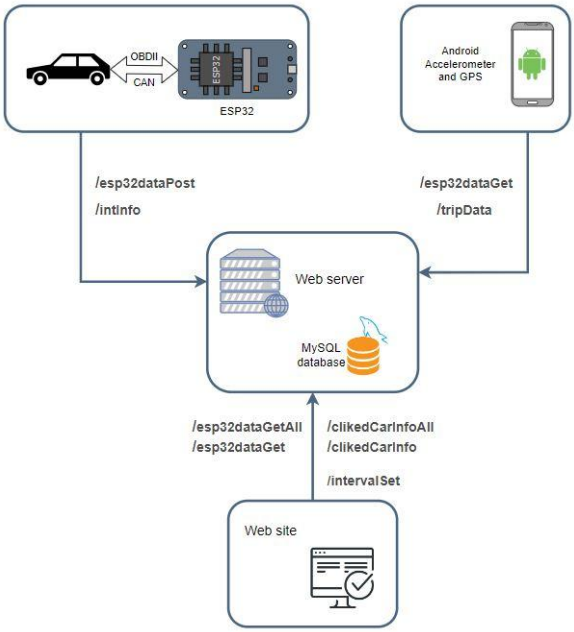
Izgled *back-end* dela veb sajta

Na slici 6. nalazi se šematski prikaz veb servera. Server prima i obrađuje zahteve koji mogu da stignu od strane:

- Android aplikacije, to jest mobilnog telefona korisnika
- *ESP32* koji prikuplja podatke sa *OBD2* porta i prosleđuje ih na server
- *Front-end*-a

Android aplikacija sa serverom komunicira u dva slučaja. Prvi slučaj je kad je završena vožena ruta (eng. *trip*) da dostavi informacije o toku vožnje i događajima, a drugi slučaj je kad želi da dobije informacije o vozilu koje je *ESP32* prethodno dostavio na server. *ESP32* sa serverom

takođe komunicira u dva slučaja, kad želi da dostavi podatke pročitane sa *OBD2* porta i kad želi da dobije interval za slanje podataka za određeno vozilo.



Slika 6. *Šematski prikaz veb servera*

4.2. *Front-end*

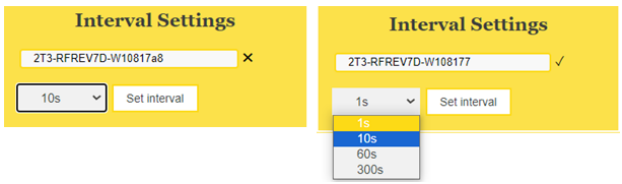
Front-end je termin koji opisuje proces kreiranja dela veb sajta koji je vidljiv korisniku i sa kojim on ima neposrednu interakciju. Implementacija grafičkog interfejsa veb sajta može se podeliti na tri celine: opis, dizajn i dodavanje funkcionalnosti stranici. U tu svrhu se koriste redom: *HTML* (eng. *Hypertext Markup Language*), *CSS* (eng. *Cascading Style Sheets*) i *JavaScript*. [5]

Izgled *front-end* dela veb sajta

Veb stranica je vizuelno podeljena na tri celine, to jest naslova:

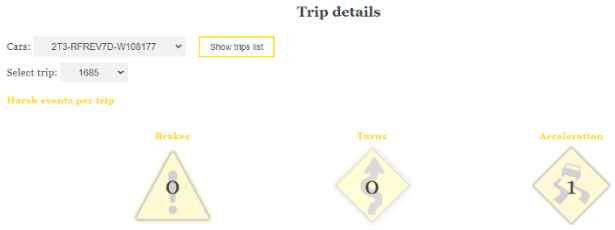
- *OBDII data* – deo o podacima sa automobila
- *Interval Settings* – deo o podešavanju intervala
- *Trip details* – deo za prikaz informacija o putovanjima

Na slici 7. prikazan je segment stranice vezan za odabir intervala. U polje je potrebno uneti *VIN* u odgovarajućem formatu. Ukoliko je format ispravno odabran iz padajućeg menija selektuje se željeni interval i pritiskom na dugme *Set interval* se sačuva.



Slika 7. *Postavljanje intervala za određeni automobil*

Centralni deo stranice zauzima deo o informacijama o rutama. Prvo je potrebno odabrati automobil, potom označiti željeno putovanje, nakon čega se dobije ispis o broju zabeleženih događaja, kao na slici 8.



Slika 8. Prikaz dela veb stranice za ispis zabeleženih događaja

Ispod informacija o događajima na veb stranici se prikazuje i mapa, sa označenim lokacijama početka i kraja putovanja, kao i tačkama u kojima se detektovani događaj desio.

5. ZAKLJUČAK

Cilj rada bila je implementacija sistema za detekciju grube vožnje. Tokom testiranja nije se naišlo na probleme ili greške. Demo snimci testiranja kao i spisak celokupne literature koja je korišćena pri implementaciji nalazi se na linku [6].

U nekim nadogradnjama sistema treba razmisliti o povećanju broja parametara koji se uzorkuju iz vozila, zatim njihovoj daljoj obradi na serverskoj strani. ESP32 delu sistema mogu se dodati senzori za lokaciju i akcelerometar i samim tim ceo sistem možemo učiniti jednostavnijim, ukloniti potrebu za *Android* aplikacijom. Server se može unaprediti dodavanjem novih funkcionalnosti gde bi korisnik mogao da bira parametre koji bi se očitavali.

6. LITERATURA

[1] https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/peripherals/twai.html, pregledano septembar 2024.

[2] https://www.csselectronics.com/pages/can-bus-simple-intro-tutorial, pregledano septembar 2024.

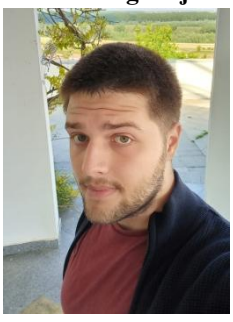
[3] https://www.csselectronics.com/pages/obd2-explained-simple-intro, pregledano septembar 2024

[4] Shah, Dhruti . Node.JS Guidebook. BPB Publications, 2018.

[5] https://www.elektronika.ftn.uns.ac.rs/umrezeni-embeded-sistemi/wp-content/uploads/sites/176/2018/03/UES-04-Front-end-i-Back-end.pdf , pregledano septembar 2024.

[6]https://drive.google.com/drive/folders/1qxiSQiPd3afc4I2gNT2fz6Po85KV1k5W?usp=drive_link

Kratka biografija:



Aleksandar Petrović rođen je 1997. godine. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnika i računarstvo – Embeded sistemi i algoritmi odbranio je 2020. god.

kontakt:
aleksandarpetrovic21897@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.38
DOI: https://doi.org/10.24867/31BE26Goronic

SISTEM ZA IDENTIFIKACIJU BILJAKA I KONTROLU USLOVA OKOLINE POMOĆU RASPBERRY PI PLATFORME
SYSTEM FOR PLANT IDENTIFICATION AND ENVIRONMENTAL CONDITION CONTROL USING RASPBERRY PI PLATFORM

Nikolina Goronić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je prikazan najidealniji algoritam za detektovanje lista biljke koristeći Raspberry Pi i Raspberry Pi kameru modul 3. Na osnovu uslikanog lista, detektuje se kojoj vrsti sobne biljke je riječ. Uz pomoć senzora DHT11 mjeri se vlažnost i provjerava se sobna temperatura prostorije, te njeno korigovanje ukoliko je potrebno paljenjem LED diode i ventilatora. Podaci se šalju sa uređaja na platformu ThingSpeak, gdje se vrši grafički prikaz vrijednosti, kao i na Node-Red, gdje je izvršena provjera funkcionalnosti.*

Ključne reči: *Raspberry Pi, DHT11 senzor, Node-Red, ThinkSpeak, digitalna slika, OpenCv, NumPi, Canny algoritam, matchShapes() funkcija*

Abstract – *This paper presents the most optimal algorithm for detecting a plant leaf using Raspberry Pi and Raspberry Pi Camera Module 3. Based on the captured leaf, the type of indoor plant is identified. With the help of the DHT11 sensor, humidity is measured and the room temperature is checked, with adjustments made if necessary by turning on the LED light and fan. The data is sent from the device to the ThingSpeak platform, where a graphical representation of the values is displayed, as well as to Node-Red, where the functionality is verified.*

Keywords: *Raspberry Pi, DHT11 sensor, Node-Red, ThingSpeak, digital image, OpenCV, NumPy, Canny algorithm, matchShapes() function*

1. UVOD

Pod pojmom "priroda" podrazumijevamo složen sistem, koji obuhvata sve što postoji, od univerzalnog do subatomskog. U okviru ovog sistema biljke imaju veliku ulogu doprinoseći održanju ravnoteže u prirodi. Jedna od najvažnijih uloga biljaka jeste obezbjeđivanje kiseonika. Posljednjih decenija, sve više ljudi spoznaje vrijednost kućnih biljaka, koje ne samo da oplemenjuju prostor svojom ljepotom, već donose i brojne koristi za zdravlje. Čiste vazduh, povećavaju vlažnost i doprinose stvaranju harmonije. Često ljudi, dobivši biljku na poklon ne znaju koja je najpogodnija sobna temperatura za opstanak iste. Ono do čega još može da dođe jeste da pomiješaju vrste biljaka, neke vrste biljaka tretiraju kao sobne što one zapravo nisu. Kako bi se suzbile ovakve greške postoji mogućnost prepoznavanja vrste biljke njihovim slikanjem, a i napravljen je spisak sobnih biljaka. Biranjem biljke dobija se njena idealna temperatura. Ovo može da zainteresuje ljude za ispravan uzgoj sobnih biljaka, da privuče neke nove potrošače koji ranije nisu imali uspjeha u uzgoju istih, da okušaju ponovo svoju „sreću“, te da podsjeti kupce i na neke zaboravljene vrste biljaka.

U ovom radu će biti prikazani različiti algoritmi i metode realizovane u programskom jeziku *Python*, koje su testirane prilikom istraživanja najadekvatnijeg algoritma za otkrivanje vrste biljke na osnovu lista, kao i prikaz toka u *Node-Red*-u, a grafika na *ThinkSpeak*-u.

2. HARDVERSKI DIO

Komponente koje su korištene prilikom realizacije rada su: *Raspberry Pi*, *DHT11 senzor*, *LED dioda*, ventilator, *Raspberry Pi kamera moduo 3* i *L298N modul*. *Raspberry Pi* je embeded računar, koji je napravljen u skladu sa *von Neumann*-ovom arhitekturom. Posjeduje sistem na čipu *BCM2837* proizvođača *Broadcom*, koji sadrži *ARM Cortex-A53* sa 4 jezgra, grafički procesor *VideoCore IV* i *RAM* memorija. 2 x 20 *GPIO* pinovi omogućuju povezivanje i rad sa drugim komponentama [1]. Dioda je povezana na pin 16 *Raspberry Pi*-ja i korišćena za signalizaciju pada temperature ispod dozvoljenog opsega. *DHT11* se koristi za mjerenje temperature i relativne vlažnosti vazduha u prostoriji. Posjeduje *NTC* termistor i kapacitivni senzor vlage. Komunikacija između modula i mikrokontrolera bazirana je na *One-wire* protokolu, koji predstavlja serijsku komunikaciju između master i jednog ili više slejv uređaja [2]. Ventilator se koristi za hlađenje komponenti, a njegovo pokretanje vrši se uz pomoć *L298N* modula, koji sadrži 11 pinova.

Ventilator je povezan na izlazne priključke OUT1 i OUT2, VCC modula je spojen sa 5V *Raspberry Pi*-ja, dok su VC i GND povezani na generator napona. Na ulazne priključke IN1 i IN2 dovodi se kombinacija visokog i niskog naponskog nivoa i na taj način se bira smer obrtanja, dok je na ulazni priključak EN doveden logički signal (logička nula i logička jedinica), na osnovu kojeg se vrši kontrola paljenja ventilatora. On je povezan sa pinom 33 *Raspberry Pi*-ja.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je dr Vladimir Rajs, van.profesor.

3. OBRADA SLIKE

Digitalna slika je elektronski zabilježena slika predstavljena u binarnom brojevnom sistemu, koja sadrži konačan broj redova i kolona piksela [3]. Pikseli su najmanji pojedinačni elementi slike, koji sadrže vrijednosti osvjetljenja date boje u svakoj specifičnoj tački.

Jedna od najpopularnijih biblioteka korišćenih za obradu slika i videa je *OpenCV*. U programskom jeziku *Python* njegovom interfejsu se pristupa putem *cv2* modula. *NumPi* je biblioteka za rad sa višedimenzionalnim nizovima (matricama) i obavlja različite matematičke operacije. U ovom radu se koristi za kreiranje matrica i za operacije nad konturama (spajanje kontura).

Prvobitna ideja je bila da se list izdvoji iz biljke i na osnovu njegovog oblika utvrdi o kojoj biljci je reč, zbog čega su izvršena testiranja na samo uslikan list ili na čitavu biljku. Algoritmi i metode koje su testirane su:

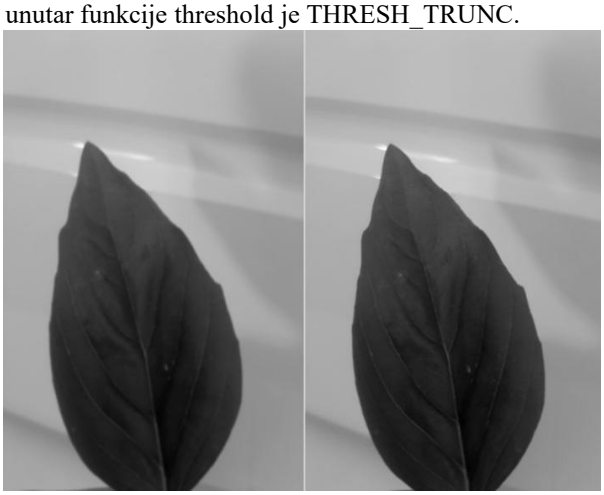
- *LoG* filter (nejasna kontura lista)
- *Sobel* (gubi se detaljnost)
- *GrabCu*t (jasno izdvajanje čitavog objekta)
- *Hough* algoritam (samo za određen oblik)
- *K-means* (neuspješnost kod sličnih boja)
- *Canny* algoritam.

Pored *Canny* algoritma (o njemu će biti više u narednoj glavi), najbolje rezultate je davao *GrabCut* algoritam, jer je dobro izdvajao biljku od sredine. Izdvajanje lista iz biljke je pokušano na osnovu boje, jer nakon djelovanja algoritma objekat se posmatra kao cijeli, pa je to jedini način za izdvajanje konture. Definisane su gornja i donja granica za boju. Koristeći gornju granicu za zelenu boju, kreira se binarna maska u kojoj su pikseli unutar opsega zelene boje označeni kao 1, a ostali kao 0. Ova maska služi za izdvajanje samo zelenih dijelova slike [4]. Ovo nije bilo uspješno, jer nemaju sve biljke istu boju listova, neke imaju svetlije, a neke tamnije. Takođe, postoji cvijeće koje ima zeleni pigment i algoritam će njih izdvojiti. Na svjetlosti i u mraku, pigment lista neće biti isti, pa samim tim ni opseg zelene boje, što znači da je jaka osjetljivost na šum i osvjetljenost.

Postoji još jedan način izolacije, a to je na osnovu oblika konture. Kako svaki list ima eliptičan tj. okrugao oblik, isprobana je izolacija upravo na osnovu toga. Na osnovu dužine i površine konture računa se kružnost, koja prikazuje koliko je oblik sličan krugu. Kružnost ima vrijednost blisku 1 za savršeno kružne objekte, a manju za izdužene ili nepravilne forme. Izračunat je aspektni odnos (odnos visina/širina) i ukoliko konture imaju aspektni odnos veći od 2.5 (odnosno visina je znatno veća od širine) i kružnost manju od 0.4, smatraju se potencijalnim listovima. Rezultat ovoga je da se pojedini listovi ne detektuju kao polukružni, dok oni koji su detektovani, ne detektuje kao cijeli list, već kao dio, pa zbog toga se ovaj algoritam ne čini pogodnim.

4. CANNY ALGORITAM

On je popularan i efikasan metod za detekciju ivica u slikama, razvijen od strane Johna F. Cannyja 1986. godine. Njegova svrha je da pronađe tačke u kojima se intenzitet boje mijenja naglo, što obično predstavlja ivicu objekta na slici. Sastoji se od pet koraka, a to su: Filtriranje šuma (Gaussian Blur), detekcija gradijenta, Non-maximum suppression, double threshold i praćenje ivica putem histerezisa (Edge Tracking by Hysteresis) [5]. Tokom realizacije postoji mogućnost modifikovanja navedenih koraka, kako bi se dobili što precizniji rezultati. Upravo na tim modifikacijama se zasniva konačno rješenje ovog rada. Nakon očitavanja slike, vrši se njena konverzija u sivu skalu pomoću cv2.cvtColor() funkcije i normalizuju se vrednosti piksela cv2.normalize() funkcijom (slika 1). Na ovaj način je slika spremna za prvi korak Canny algoritma, a to je Gaussian Blur. Uklanjanjem šumova potrebno je da se ona threshold-uje, postavljajući odgovarajući prag sa kojim se porede svi pikseli na slici. Izabrana tehnika unutar funkcije threshold je THRESH_TRUNC.



Slika 1. *Slika uslikanog lista nakon blurovanja (lijevo) i normalizacije (desno)*

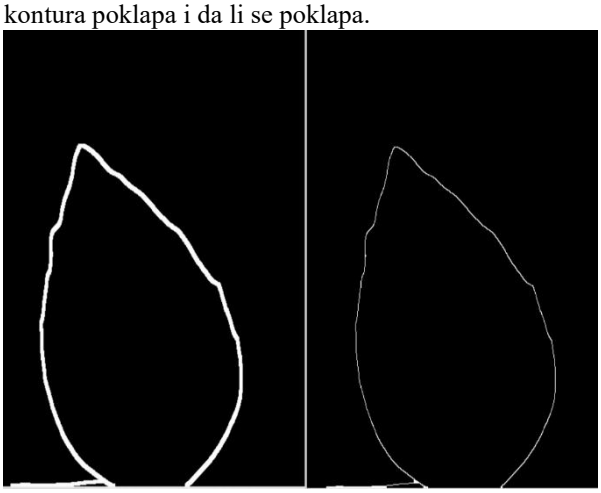
Sljedeći korak je detekcija ivica i crtanje kontura, (slika 2), koje su na nekim listovima isprekidane, dok su na nekim savršene. Ideja je bila da se iz skupa isprekidanih kontura izdvoji najveća i uporedi sa bazom biljaka, međutim bezuspešno.

Slika 2. *Slika uslikanog lista nakon thresh_trunc tehnike (lijevo) i detektovanih ivica (desno)*

Jako su velike greške, kao i razlike između iste vrste listova. Upravo iz tog razloga je izvršeno sabiranje konturica lista i dobijanje jedne konture. Prije nego što se izvrši sabiranje kontura, bilo je potrebno da se izvrši provjera da li je kontura potpuno zatvorena (konture na

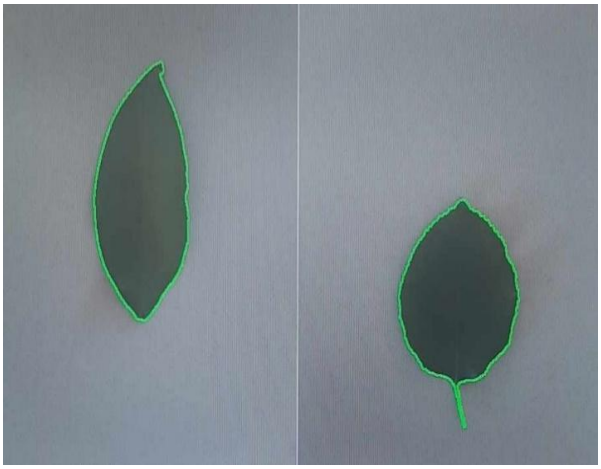
nekim listovima su savršene) ili kada je kontura skoro zatvorena, jer se desi da se uslika pola lista. Posmatrale su se krajnje tačke koje su najbliže i na osnovu toga se izdvajala ta kontura. Ovo nije dalo uspješne rezultate, jer su se pojavili slučajevi kada konture unutar pora imaju manju razliku u tačkama.

Pošto prethodni koraci nisu davali najbolje rezultate, nego prosječne, bilo je potrebno izvršiti poboljšanje ivica. Zbog toga su iskorištene dve funkcije cv2.dilate() i cv2.erode() nakon crtanja ivica, a pre crtanja konture. Dilatacijom su proširene ivice i pomaže u popunjavanju praznina, dok erozija „stanjuje“ ivice. Ovo je doprinijelo poboljšanju strukture i detektovane su bolje konture (slika 3), bez prekida. Kada se odredi oblik konture uslikanog lista, potrebno je da se proveri sa kojim se od kontura poklapa i da li se poklapa.



Slika 3. *Slika uslikanog lista nakon dilatacije (lijevo) i erode funkcije (desno)*

Funkcija cv2.matchShapes() u OpenCV-u služi za upoređivanje oblika, odnosno za mjerenje sličnosti između kontura [6]. Koristi Huove momente, koji su invarijantni na translaciju, skaliranje i rotaciju, što znači da su otporni na promjene u veličini i položaju oblika. Ova funkcija prolazi kroz bazu kontura svih listova i vraća numeričku vrijednost koja predstavlja koliko su konture slične. Manje vrijednosti (bliže nuli) označavaju veću sličnost između kontura, dok veće vrijednosti označavaju da su konture znatno različite. Provjerava se koja je najmanja sličnost između biljaka i da li je manja od 1.



Slika 4. *Konture dva različita lista*

Ukoliko je uslov ispunjen, ispisuje se koja je vrsta biljke. Kao što je rečeno u prethodnom poglavlju pre korišćenja dilatacije i erozije, ova funkcija je davala prosječne rezultate. Većinom je otkrivala ispravnu vrstu biljke, međutim dešavalo se da drastično pogriješi, kao što se može videti na slici 4. Lijevi list se poklapao sa konturom desnog lista 0,48, što je prevelika vrijednost. Nakon uvođenja navedenih funkcija, tačnost je postala bolja i najmanja vrednost se dobija za vrstu biljke kojoj pripada.

5. NODE-RED I THINGSPEAK

Kao što je već rečeno u integrisanom razvoju okruženju, nazvanom *Thonny*, napisan je odgovarajući kod za implementaciju senzora *DHT11* i komunikaciju između odgovarajućih platformi, kao i rad sa kamerom i algoritmom za obradu slike koristeći programski jezik *Python*. Kako bi *DHT11* senzor radio ispravno, potrebno je instalirati biblioteku Adafruit_Python_DHT.

Korišćenje funkcije *Adafruit_DHT.read_retry* čitaju se podaci sa senzora. Vrijednosti sa senzora se šalju na *ThingSpeak* platformu (*IoT* analitička platforma, koja omogućava vizuelizaciju i analizu podataka) i *Node-*RED koristeći *HTTP* protokol, koji se zasniva na principu server-klijent. Korisnik šalje zahtev, a server odgovara na taj zahtev.

Node-RED Flow (tok) se odnosi na povezivanje i sekvenciranje različitih ulaznih, izlaznih i procesnih čvorova unutar *Node-RED* platforme [7]. Svaki čvor unutar toka obavlja jedinstven i specifičan zadatak. Kada se podaci prenose do čvora, čvor ih obrađuje prema svojoj naznačenoj funkciji, prije nego što ih proslijeđuje sljedećem čvoru u toku.

Potrebno je odabrati jedan od dva button-a, koji predstavljaju način na koji korisnik želi da se utiče na uslove u prostoriji, ručno ili automatski. Ukoliko se klikne na taster “Automatski” pojavljuje se dropdown čvor i dobija se lista biljaka.

Klikom na ime biljke koja se nalazi u prostoriji, iskače message (show notrification) sa optimalnim opsegom temperature, koji se dalje šalje na join čvor na koji stiže i temperature sa senzora. Kada pristugnu oba podatka na join čvor, prelazi se na sljedeći čvor, gdje se provjerava koji je podatak prvi stigao, a koji drugi. Nekada se desi da prvo stigne podatak sa senzora, a nekada sa dropdowna. Ukoliko je podatak string, stigla je vrijednost sa *dropdown*-a, a ukoliko je number sa senzora Pristigle vrijednosti se smiještaju u promjenljive temperaturaDropdowna i temperaturaPosta. Potrebno je razdvojiti opseg temperature iz dropdowna, na minimalnu i maksimalnu vrijednost, koje su u dropdownu zapisane u obliku: minTemperatura-maxTemperatura.

Funkcija *split* je korištena za razdvajanje minimalne i maksimalne vrijednosti temperature. Ukoliko je temperatura senzora manja od minimalne vrijednosti dropdowna, pali se diode i gasi se ventilator, a ukoliko je veća od maksimalne vrijednosti temperature, gasi se diode i pali se ventilator. Kako *join* čvor uzima bilo koja dva podatka koja mu stižu, u funkciji je naznačeno da je

to pogrešno i ne uzima se u obzir prilikom upoređivanja temperatura.

Ukoliko je izabran taster “Rucno”, pojavljuje se padajućo meni sa biljkama, njihovim opsezima i poruka da je neophodno uporediti temperature sa senzora i opseg idealne temperature. Ukoliko je neophodno upaliti diodu ili ventilator, moguće je pomoću 2 *switch* čvora, koji kada su on tj.1, pale diodu ili ventilator, a off, gase.

6. ZAKLJUČAK

Uspješno je realizovana tema koristeći *DHT11* senzor za očitavanje temperature i vlažnosti, čije se vrijednosti prikazuju na *ThingSpeak* platformi. Uspješno je primljena vrijednost temperature i u Node- RED-u, kao i njeno poređenje sa opsegom idealne temperature određene biljke i paljenje *LED* diode ili ventilatora u skladu sa zadatim uslovima. Pronađen je odgovarajući algotiram za detekciju oblika lista nakon što je uslikan pomoću Raspberry Pi kamere. Testirane su slike, koje su uslikane kamerom sa mobilnog telefona, čime je potvrđena mogućnost korišćenja algoritma pri različitim uslovima.

Postoje mogućnosti poboljšanja rada, a to su:

- Smanjivanje/povećavanje jačine ventilatora u skladu sa neophodnom temperaturom
- Korišćenje nekih drugih aktuatora za signalizaciju, u slučaju kvara jednog od korišćenih
- Praćenje osvjetljenosti biljke uz pomoć senzora (npr. *BH1750*)
- Neke biljke zavise od godišnjih doba, pa im optimalne temperature variraju zimi i ljeti, pa bi ideja bila da se odabirom godišnjeg doba izlistaju idealne temperature u tom period
- Korišćenje *lm7805*, tranzistora ili mosfetova prilikom stabilizacije napona

Problemi prilikom realizovanja projekta:

- Nemogućnost istovremenog dolaska podatka sa senzora i sa dropdowna, kao i ne čuvanje trenutne vrijednosti, dok ne dođe nova (ne pomažu delay, ni *time.sleep(),* ni if i wait naredbe). Zato je korišten join čvor, ali mora se paziti na vrijeme čekanja
- U slučaju da se više podataka dovodi na drugi čvor, voditi računa koji podatak prvi stiže i uzeti u obzir sve mogućnosti, posebno ako su istog tipa
- Nemogućnost izdvajanja lista iz biljke koristeći algoritme za izdvajanje lista na osnovu boje in a osnovu oblika
- Ukoliko se dva lista drastično razlikuju, sličnost će im biti velika, a ukoliko se manje razlikuju, sličnost im je manja. Samim tim postoji mogućnost greške ukoliko se list biljke ne nalazi u bazi, a sličnost im je ispod 1. Ako se list biljke nalazi u bazi, neće biti greške.
- Nemogućnost prepoznavanja istih listova prilikom različitih skaliranja (ovo je riješeno).

11. LITERATURA

[1] Ivan Mezei (2016) Računarska elektronika, praktikum laboratorijskih vežbi

[2] https://components101.com/sensors/dht11-temperature-sensor (pristupljeno u septembru 2024.).

[3] https://dsp.etfbl.net/multimediji/2017/09_slika.pdf (pristupljeno u septembru 2024.).

[4] https://www.geeksforgeeks.org/filter-color-with-opencv/ (pristupljeno u septembru 2024.).

[5] https://towardsdatascience.com/canny-edge-detection-step-by-step-in-python-computer-vision-b49c3a2d8123 (pristupljeno u septembru 2024.).

[6] https://www.tutorialspoint.com/how-to-match-image-shapes-in-opencv-python*2013* (pristupljeno u septembru 2024.).

[7] https://en.wikipedia.org/wiki/Node-RED (pristupljeno u septembru 2024.)

- **Kratka biografija:**

Nikolina Goronić rođena je u Prijedoru, gdje završava osnovnu i srednju školu. Na Fakultetu tehničkih nauka odbranila je 2023. diplomski rad iz oblasti Embeded sistema i algoritama, na temu: “Projektovanje i verifikacija Linear Congruential Generator hardverskog akceleratora”.

kontakt:
nikolinagoronic@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 621.38
DOI: https://doi.org/10.24867/31BE28Panic

UTICAJ PARAZITNIH EFEKATA PCB SUPSTRATA NA SIGNALE LED DRAJVERA

INFLUENCE OF PCB SUBSTRATE PARASITIC EFFECTS ON LED DRIVER SIGNALS

Milica Panić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu analiziran je uticaj parazitnih efekata štampane ploče (PCB) na signale LED drajvera, koji se koriste u automobilskim displejima. Posmatrani su signali mase, kominikacioni signali i signali napajanja LED drajvera u različitim tačkama štampane ploče. Na osnovu rezultata merenja, predložen je redizajn PCB-a, da bi se postiglo poboljšanje kvaliteta signala LED drajvera.*

Ključne reči: *EMC, parazitni efekti, LED drajver, PCB dizajn*

Abstract – *This paper presents the influence of the printed circuit board (PCB) parasitic effects on LED driver signals used in automotive displays. Ground signals, communication signals and power signals of the LED drivers at different points have been measured. Based on the measurement results, a redesign of the PCB was done to achieve an improvement in the signal quality of the LED driver.*

Keywords: *EMC, parasitic effects, LED drivers, PCB design*

1. UVOD

Napretkom tehnologije i minimizacijom komponenti, pojavljuju se novi izazovi u kontroli parazitnih efekata na štampanim pločama (PCB) poput kapacitivnosti, induktivnosti i otpornosti provodnih linija, koji mogu negativno uticati na kvalitet signala. LED tehnologija postala je standard u raznim primenama, od automobilskih panela do reklamnih displeja.

U oblastima kao što su telekomunikacije, automobilska elektronika i medicinska oprema, parazitni efekti mogu značajno degradirati performanse i povećati greške u prenosu podataka. Na primer, u telekomunikacijama, mali nivo šuma ili kašnjenja može ometati kvalitet signala, dok u automobilskoj elektronici ti efekti mogu ugroziti sigurnost vožnje. Takođe, u medicinskim uređajima, pouzdanost i tačnost signala su ključni za pravilnu dijagnozu i tretman pacijenata. Zato dizajn štampanih ploča mora biti optimizovan kako bi se obezbedio stabilan rad i eliminisali uticaji parazitnih efekata na kvalitet signala [1].

NAPOMENA
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Mirjana Damnjanović, red. prof.

Da bi se razmotrili ovi neželjeni efekti, analizirani su signali komercijalno dostupnog LED drajvera TLC6C5748-Q1, koji se koristi u automobilskoj industriji za navigacione displeje ili instrument table [2]. Izmereni su signali LED drajvera u nekoliko različitih tačaka, i to komunikacioni signali SIN (*Serial In*), signali napajanja VLED (*Voltage LED*) i signali mase GND (*Ground*). Da bi se smanjio uticaj parazitnih efekata, razmatrana su tri različita lejauta PCB-a.

2. OPIS SISTEMA I METODOLOGIJA ISTRAŽIVANJA

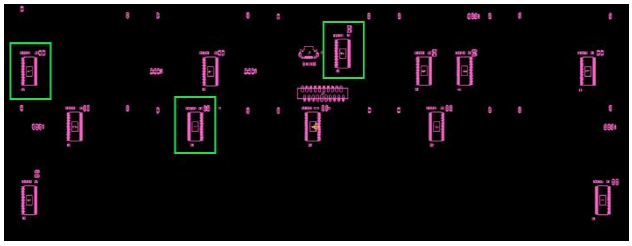
Analizirana ploča deo je sistema koji predstavlja displej za automobile. Ceo sistem se sastoji od dve štampane ploče: glavne (Main PCB) i ploče pozadinskog osvetljenja (LED PCB). Glavna ploča sadrži naponski konvertor koji obezbeđuje napajanje za LED diode, kao i mikrokontroler koji generiše i upravlja komunikacionim signalima SPI (*Serial Peripheral Interface*) protokola. Ove dve ploče međusobno komuniciraju preko dva kabla. Preko *micro-match* kabla prenosi se GND signal i napajanje za diode, VLED, dok se preko FFC (*Flexible Flat Cable*) kabla prenose komunikacioni signali (SPI).

LED PCB ploča, sadrži ukupno dvanaest drajvera, od kojih su testirana tri, raspoređena na različitim pozicijama, kako bi se utvrdilo kako parazitni efekti, kao što su dužina i geometrija provodnih linija, utiču na signale LED drajvera.

Testirani su drajveri u centru ploče, levo od centra i na ivici sa leve strane, kao što je prikazano na slici 1 (uokvireni zelenom linijom). Cilj testiranja je bio da se ispita ponašanje signala LED drajvera raspoređenih na različitim pozicijama.

3. LED DRAJVER TLC6C5748-Q1

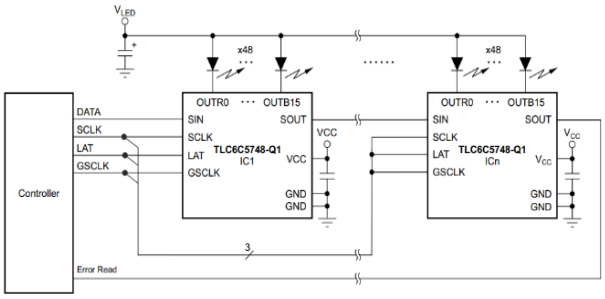
Drajver TLC6C5748-Q1, proizvođača Texas Instruments [2], može da se koristi za kontrolu velikog broja LED dioda. Ovaj automobilski (*automotive*) drajver omogućava



Slika 1. *Prikaz pozicije testiranih LED drajvera na LED PCB ploči*

kontrolu osvetljenja uz minimalno stvaranje toplote i elektromagnetskih smetnji. Drajver ima ukupno 48 izlaznih kanala. Svaki kanal ima nezavisno podešavanje osvetljenosti, čime se obezbeđuje veći svetlosni izlaz, jer su sve diode aktivne istovremeno. Kontrolu intenziteta svetlosti pruža 16-bitna PWM kontrola, sa 65536 koraka osvetljenosti.

Više ovakvih drajvera se može povezati redno (*daisy chain application*), čime se omogućava napajanje većeg broja LED sijalica sa manjim brojem portova kontrolera (slika 2) [2]. Za komunikaciju koristi SPI protokol, koji zahteva SDO (*Serial Data Out*), SIN (*Serial In*), SCK (*Serial Clock*) i LAT (*Latch*) pinove.



Slika 2. *Daisy chain povezivanje dva TLC6C5748-Q1 LED drajvera, za napajanje većeg broja LED sijalica [2]*

4. PARAZITNI EFEKTI PCB SUPSTRATA

Parazitni efekti odnose se na neželjene efekte koji nastaju na PCB supstratu zbog fizičke strukture i međusobnih veza pojedinih delova sistema. Ovi elementi uključuju parazitnu kapacitivnost, induktivnost i otpornost; naročito počinju da dominiraju na višim frekvencijama.

4.1. Parazitna kapacitivnost

Parazitna kapacitivnost je neželjena kapacitivnost koja može nastati između dva provodnika na štampanoj ploči, kao i unutar samih elektronskih komponenti. Obično je reda pikofarada (pF).

Parazitne kapacitivnosti mogu uzrokovati neželjene efekte, kao što su:

- neželjene oscilacije,
- preslušavanje ili
- smanjenje brzine prelaznih procesa [3].

4.2. Parazitna induktivnost

Parazitna induktivnost nastaje usled magnetnog polja koje stvaraju provodnici kroz koje teče struja. Na PCB-u, struja koja prolazi kroz provodnike stvara magnetsko polje. Ovo se opisuje preko samoinduktivnosti provodnika. Takođe, ukoliko postoje dva provodnika na malom rastojanju, može doći do neželjene sprege, što se opisuje preko međusobne induktivnosti.

Parazitna induktivnost može značajno uticati na kvalitet signala, posebno na višim frekvencijama i za struje većeg intenziteta, stvarajući time veću induktivnu reaktansu i elektromagnetsko polje.

Parazitna induktivnost obično je u opsegu nanohenrija (nH) i može dovesti do sledećih problema:

- povećanog elektromagnetskog zračenja,
- neželjenih oscilacija ili
- kratkotrajnih prebačaja i podbačaja napona, tj. visokih i niskih naponskih impulsa (*overshoots, undershoots*) [3].

4.3. Parazitna otpornost

Svaki provodna linija na PCB-u ima određenu otpornost, koja raste povećanjem dužine i smanjenjem širine (tj. poprečnog preseka) linije. Kako raste potreba za kompaktnijim štampanim pločama, gustina komponenti se povećava, što rezultira tanjim i dužim vodovima, a samim tim i većim padom napona zbog parazitne otpornosti, što se može negativno odraziti na integritet signala.

5. ANALIZA SIGNALA LED DRAJVERA

Merna postavka koja je korišćena za testiranje LED drajvera se sastoji od:

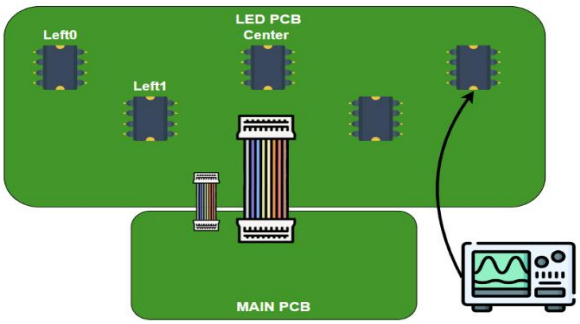
- osciloskopa Rohde & Schwarz RTO6 [4],
- napajanja AIM-TTI CPX400D [5] i
- odgovarajućih kablova.

Testiranje je sprovedeno na 25 °C, sa napajanjem od 13,5 V i strujom od 6,9 A (574 dioda × 12 mA).

Posmatrani signali su:

- SIN (pin 1) – komunikacioni signal preko SPI protokola (0 V ili 3,3 V),
- VLED – napajanje za LED (napon na anodama),
- GND (pin 28) – lokalni GND drajvera.

Pomoću softvera, impulsno širinska modulacija PWM (*Pulse Width Modulation*) je podešena na 10 %. Ova vrednost je izabrana jer pri 100 % ispune neće doći do pojave impulsa, već će signal biti kontinualan, što otežava analizu. Takođe, pri visokom procentu ispune (tj. pri velikim osvetljenjima), prekomerna svetlost može biti naporna za oko zbog velikog broja dioda.



Slika 3 *Grafički prikaz merne postavke za testiranje analiziranog sistema korišćenjem osciloskopa RTO6*

5.1. Analiza GND signala

Korišćenjem osciloskopa RTO6 izmereni su signali mase u četiri tačke PCB-a (slika 3). Prvi, Main PCB GND signal, snimljen je na masi izlaznog kondenzatora *buck* konvertora na glavnoj ploči. Na osnovu rezultata merenja (slika 4, žuta boja), može se zaključiti da je ovo stabilan, referentni signal, sa maksimalnom amplitudom od 35,07 mV, što je približno jednako očekivanoj vrednosti napona od 0 V.

Međutim, ovo neće biti slučaj i za masu u preostale tri tačke. Što je drajver udaljeniji od glavnog konektora, integritet signala se sve više pogoršava. Drugi signal mase, prikazan na slici 4 zelenom bojom, izmeren je u tački Center LDRV GND, na prvom drajveru, koji je najbliži

Main PCB-u. Signal ima uočljiva izobličenja, amplitude vrednosti do 254,45 mV.

Treći signal, Left1 LDRV GND, izmeren je na masi osmog drajvera, sa amplitudom smetnji od 398,81 mV (slika 4, plava boja), dok je četvrti signal, Left0 LDRV GND sa poslednjeg, najudaljenijeg drajvera, sa najvećim smetnjama, koje dostižu čak 578,00 mV (slika 4, crvena boja).

Frekvencija izmerenih smetnji iznosi 122 Hz, što odgovara frekvenciji PWM signala koji se koristi za upravljanje LED diodama. Ove smetnje su rezultat superponiranja usled preslušavanja, što uzrokuje promene naponskih nivoa.

Prisutne smetnje na GND signalu mogu dovesti do izobličenja i drugih signala, merenih u odnosu na GND. Na primer, nizak logički nivo može biti povišen, a logički visok nivo smanjen usled pogrešne vrednosti signala na masi, što može dovesti do pojave greške u komunikaciji i obradi signala.

5.2. Analiza SIN signala

Na slici 5 prikazan je oscilogram sa tri komunikaciona SIN signala. Prvi signal, obeležen kao Center LDRV SIN (zelena boja), predstavlja izmereni SIN signal sa centralnog, prvog drajvera. Drugi signal, Left1 LDRV SIN (plava boja), je signal osmog drajvera po redu u komunikaciji. Treći signal, Left0 LDRV SIN (crvena boja) predstavlja signal poslednjeg, dvanaestog drajvera.

Kao i u slučaju GND signala, mogu se uočiti smetnje i na izmerenim komunikacijskim SIN signalima. Amplituda superponiranih smetnji, a time i ukupna amplituda signala, rastu kako se signal udaljava od glavnog konektora. Amplituda superponirane smetnje na SIN signalu prvog drajvera iznosi 170,33 mV, drugog drajvera 217,05 mV, dok poslednji drajver dostiže amplitudu od 540,7 mV.

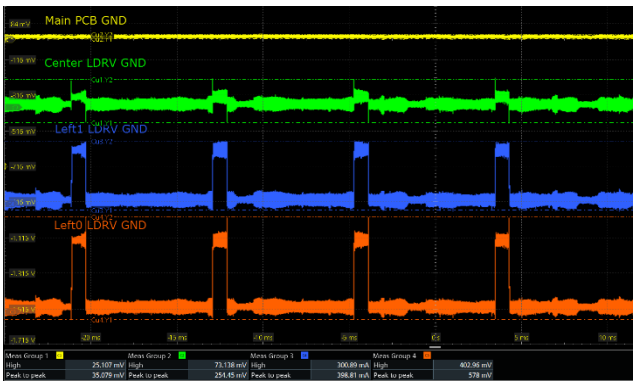
Prema katalоškim podacima drajvera [2], minimalna vrednost za registrovanje visokog naponskog nivoa je 2,31 V, a maksimalna za nizak nivo je 1,00 V. Dakle, ukoliko bi neželjeni impulsi imali amplitudu veću od 1 V, moglo bi se desiti da drajver pogrešno registruje visoki nivo umesto niskog, što može dovesti do gubitka podataka i grešaka u komunikaciji.

5.3. Analiza VLED signala

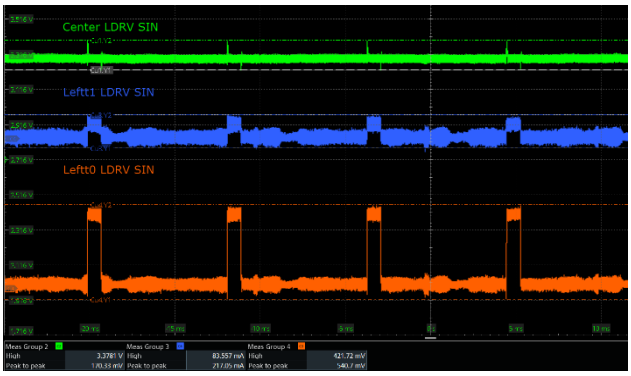
Imajući u vidu priključen broj dioda, *buck* konvertor je podešen da generiše izlazni napon VLED=7,2 V, [2]. Na slici 6 su prikazani rezultati merenja signala napajanja VLED u nekoliko različitih tačaka. Prvi signal, Main PCB VLED (žuta boja), predstavlja signal sa Main PCB-a i amplituda superponirane smetnje iznosi 305,58 mV.

Drugi signal, Center LDRV VLED (zelena boja), dolazi sa centralnog drajvera, i ima superponirane smetnje amplitude 587,65 mV. Left1 LDRV VLED (plava boja) je napon osmog drajvera, sa amplitudom neželjenog signala od 603,14 mV, dok signal poslednjeg drajvera, Left0 LDRV VLED (crvena boja), ima amplitudu od 612,96 mV.

Dakle, može se primetiti da nema značajnije razlike u amplitudi između VLED signala posmatrana tri drajvera i da svi sadrže superponirane smetnje amplitude oko 600 mV. Da bi se ovo objasnilo, potrebno je posmatrati dizajn lejauta. Naime, na gornjoj strani LED PCB-a, napajanje VLED je raspoređeno po celoj površini, dok se naponi na katodama dioda dovode preko vija (odnosno, ne postoje provodne linije za dovod napajanja VLED na kojima se može javiti pad napona). Zbog toga, izmereni signali su približno jednaki.

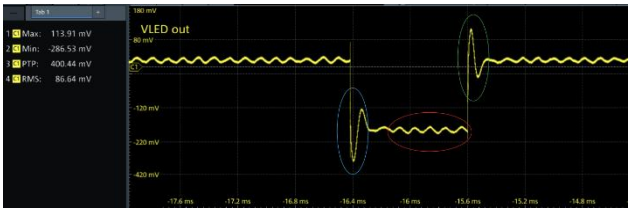


Slika 4. *Oscilogram GND signala izmerenih u četiri tačke (Main PCB, Center LDRV GND, Left1 LDRV GND i Left0 LDRV GND)*



Slika 5. *Oscilogram SIN signala izmerenih sa centalnog, osmog i dvanaestog po redu drajvera u kominikaciji*

Slika 6. *Oscilogram VLED signala izmerenih sa Main PCB-a i centalnog, osmog i dvanaestog drajvera (Center LDRV GND, Left1 LDRV GND i Left0 LDRV GND)*



Slika 7. *Oscilogram sa uvećanim prikazon Main PCB VLED signala*

Na slici 7 prikazan je oscilogram signala Main PCB VLED, gde se mogu uočiti uticaji parazitnih efekata. Oscilacije (*ringing*), uokvirene crvenom bojom, rezultat su kombinacije parazitne induktivnosti i kapacitivnosti koje formiraju rezonantni krug. Deo signala uokviren plavom bojom predstavlja *undershoot*, dok je zelenom obeležen *overshoot*. Ove pojave nastaju usled naglih promena opterećenja. Isključenje diode dovodi do smanjenja struje, time i *overshoot*-a, dok povećanje struje izaziva *undershoot*.

5.4. Redizajn lejauta za poboljšanje odziva signala

Vod od glavnog konektora na LED PCB-u do GND pina krajnjeg LED drajvera, na kojem su izmerene najveće amplitude smetnji, dužine je 164,68 cm. Kao što je prethodno prikazano, parazitni efekti PCB-a uzrokuju različite smetnje na signalima GND, VLED i SIN, koje postaju sve izraženije povećanjem dužine provodnih linija.

Zbog toga je urađen redizajn lejauta tako da je smanjena dužina, a povećana širina provodnih linija (čime se smanjuju parazitne otpornosti i induktivnosti). Umesto provodne linije prečnika 0,125 mm, za povezivanje lokalnog GND drajvera sa stabilnim GND na Main PCB-u, korišćena su dva kabela:

1. kabl dužine 31 cm, prečnika 0,42 mm, poprečnog preseka 0,14 mm², podužne otpornosti 140 mΩ/m;

2. kabl dužine 31 cm, prečnika 1,38 mm, poprečnog preseka 1,50 mm², podužne otpornosti 10 mΩ/m.

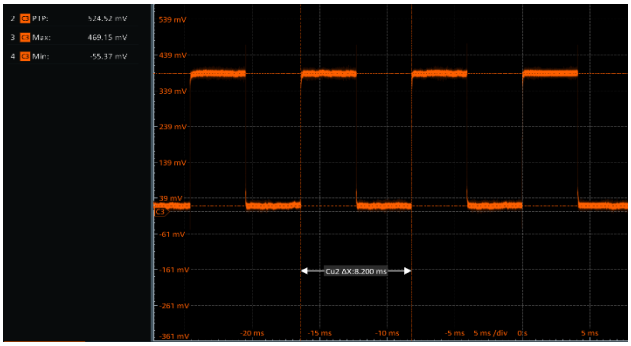
Na slici 8, prikazan je GND signal krajnjeg, dvanaestog LED drajvera pre redizajna, sa neželjenim naponskim impulsom amplitude 524,52 mV, dok je na slici 9 prikazan isti signal uz korišćenje kabla prečnika 0,42 mm (dakle, oko tri puta širi od početnog provodnika). Nakon redizajna, amplituda smetnji je smanjena na 217,72 mV.

Na slici 10 prikazan je GND signal krajnjeg, dvanaestog LED drajvera uz korišćenje drugog kabla, još većeg prečnika od 1,38 mm, dakle, jedanaest puta veći od početnog. Izmerena amplituda smetnji je smanjena još više nego u prvom slučaju, na 190,91 mV. Dakle, optimizacija lejauta povećanjem poprečnog preseka i smanjenjem dužine provodnika dovodi do smanjenja parazitnih efekata i značajnog smanjenja neželjenog pada napona.

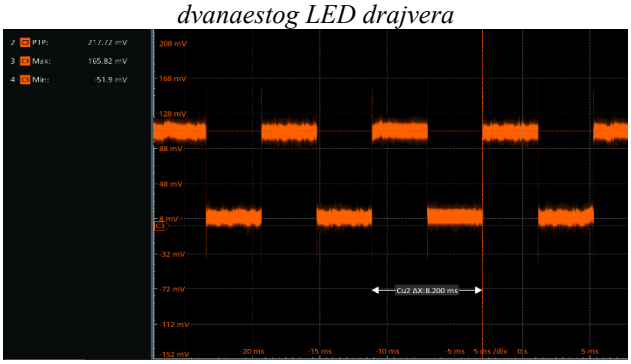
6. ZAKLJUČAK

U ovom radu analiziran je uticaj parazitnih efekata na signale LED drajvera u automobilskim displejima. Merenja pokazala su da signali LED drajvera, GND, SIN i VLED sadrže određeni nivo superponiranih smetnji, koje se manifestuju kao nepoželjni naponski impulsi, i koje postaju značajnije povećanjem dužine vodova. Takođe, uočeni su efekti poput *ringing*-a, *undershoot*-a i *overshoot*-a, koji su posledica parazitne induktivnosti i kapacitivnosti.

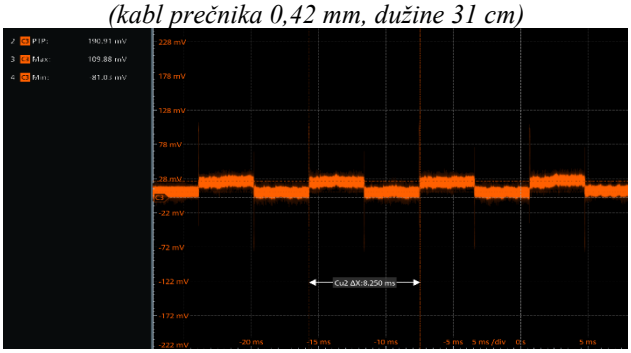
Merenjima je povrđeno da redizajn PCB-a može značajno smanjiti superponirane smetnje, i to prvenstveno kroz smanjenje dužine vodova i povećanje njihovog poprečnog preseka. U budućim istraživanjima, akcenat može biti na primeni novih materijala i tehnika za smanjenje negativnih efekata parazitnih elemenata (tj. korišćenje komponenti sa poboljšanim fizičkim svojstvima, kao i ispitivanje uticaja izbora komponenti kroz poređenje *through hole* i SMD komponenti u istom dizajnu.



Slika 8. *Oscilogram GND signala sa početnim lejautom dvanaestog LED drajvera*



Slika 9. *Oscilogram GND signala dvanaestog LED drajvera nakon redizajna (kabl prečnika 0,42 mm, dužine 31 cm)*



Slika 10. *Oscilogram GND signala dvanaestog LED drajvera nakon redizajna (kabl prečnika 1,38 mm, dužine 31 cm)*

7. LITERATURA

[1] H. Grace, „The impact on Parasitics in PCB Design“, PCB007 Magazine, August 2024, pp: 66-68.

[2] Texas Instruments: „LED drajver TLC6C5748-Q1“, 2020. Dostupno na: https://www.ti.com/product/TLC6C5748-Q1. Pristupljeno: sept. 2024.

[3] C. Coombs, H. Holden: „Printed Circuits Handbook, seventh edition“, McGraw-Hill Education, 2016.

[4] Rohde&Schwarz, Osciloskop RT06. Dostupno na: https://www.rohde-schwarz.com/brochure-datasheet/rto6/ Pristupljeno: sept. 2024.

[5] AIM-TTI, Laboratorijsko napajanje CPX400D. Dostupno na: https://www.aimtti.com/go/cpx/index.php?p=index-cpx400d Pristupljeno: sept. 2024.

Kratka biografija:

Milica Panić rođena je u Banja Luci 1999. god. Master rad na Fakultetu tehničkih nauka iz oblasti Primenjena elektronika odbranila je 2024.god. Kontakt:panicmilica1406@gmail.com

Зборник радова Факултета техничких наука, Нови Сад

UDK: 621.317
DOI: https://doi.org/10.24867/31BE29Bosancic

ДИЗАЈН АУТОМАТИЗОВАНИХ ТЕСТ СИСТЕМА ЗА ФУНКЦИОНАЛНО ТЕСТИРАЊЕ У СЕРИЈСКОЈ ПРОИЗВОДЊИ

DESIGN OF AUTOMATIC TEST EQUIPMENT FOR FUNCTIONAL TESTING IN SERIAL PRODUCTION

Коста Босанчић, *Факултет техничких наука, Нови Сад*

Област – МЕРНИ СИСТЕМИ

Кратак садржај – *Процеси верификације серијске производње штампаних плоча технологијом површинске монтаже. Смернице за дизајнирање аутоматске тест опреме за функционално тестирање који доводе до побољшања ефикасности и смањења трошкова у серијској производњи штампаних плоча.*

Кључне речи: *Серијска производња, тестирање штампаних плоча у производном процесу, аутоматизована тест опрема, тест системи*

Abstract – *Automated processes for verification of surface mount assembly production process. Guidelines for designing automatic test equipment that lead to improved efficiency and cost reducion in serial producition.*

Keywords: *Serial production, verification of printed circuit boards, automatic test equipment, test systems*

1. УВОД

У савременој електронској индустрији, серијска производња штампаних плоча представља сложен и вишестепени процес. У току производње штампаних плоча технологијом површинске монтаже низ машина раде повезано како би омогућили брз и аутоматизован процес производње. Производна линија састоји се од узастопних процеса: наношење пасте за лемљење, постављање компоненти, процес лемљења унутар пећи и оптичка инспекција. Технике као што су *In-Circuit, Flying Probe, Boundary Scan* и *End-of-Line* тестирање представљају кључне методе за идентификацију и исправљање потенцијалних грешака које могу настати током производње штампаних плоча.

Циљ је да се кроз анализу аспеката дизајнирања аутоматизованих тест система пружи свеобухватан преглед најбољих практичних решења и пракси које инжењери аутоматизоване тест опреме користе да би омогућили ефикасну и поуздану производњу. На тај начин, рад ће пружити свеобухватан увид у интеграцију аутоматизације у процес тестирања, чиме ће се омогућити боље разумевање изазова и решења која су неопходна за постизање високог нивоа квалитета у серијској производњи штампаних плоча.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији је ментор био др Платон Совиљ

2. МЕТОДЕ ТЕСТИРАЊА У СЕРИЈСКОЈ ПРОИЗВОДЊИ

Циљ тестирања у серијској производњи је потврда функционалности уређаја и верификација производне линије. Функционално тестирање у производњи нема за циљ верификацију дизајна, већ процес производње.

2.1. *In Circuit* тестирање

In-Circuit тестирање је кључна техника која се користи у производњи електронике како би се осигурала исправност и квалитет штампаних плоча. *In-Circuit* је дизајниран како би верификовао широк спектар грешака, укључујући кратке спојеве, отворене везе и погрешне вредности компоненти. Идентификацијом наведених проблема у раној фази производње, *In-Circuit* системи помажу у спречавању да дефектне плоче наставе да троше ресурсе. *In-Circuit* тестер се обично састоји од две главне целине: тест адаптера и аутоматске тест опреме. Тест адаптер или популарно назван "*Bad Of Nail*", састоји се од иглица које омогућују директан физички приступ тест локацијама на штампаној плочи. Аутоматизована тест опрема примењује тест векторе, прима сигнале и контролише процедуру тестирања. Тест иглице ступају у контакт са посебно одређеним тестним тачкама на штампаној плочи. Неке од ових тестних тачака се природно налазе на плочи, као што је лемна страна код *Through-Hole* компоненти. Код *Surface Mount* компоненти, потребно је предвидети и поставити тестне тачке током дизајна [1].

2.2. *Boundary Scan* тестирање

Напредак у технологији интегрисаних кола и паковању уређаја довео је до минијатуризације чипова, самим тим приступ чипу ради тестирања постао је ограничен. Да би превазишли проблеме минијатуризације група инжењера удружила је снаге како би пронашла решење теситрања сложених штампаних плоча и стандардизовала приступ тестирању. Овај приступ постао је *IEEE* стандард 1149.1. познатији као *Boundary Scan.* Увођењем *Boundary Scan*-а долази до смањења броја тестних тачака потребних на штампаној плочи, што резултира једноставнијим дизајном и мањим трошковима за израду тест адаптера, смањењем времена или потенцијално елиминисањем *In-Circuit* тестирања. За *Boundary Scan* тестове у чипове се додаје посебна логика која се састоји од *Boundary Scan* ћелије и регистра а њима се приступа преко *Test Access Port-*

а. *Boundary Scan* ћелија се поставља између *core* логике и портова. У нормалном режиму, ове ћелије не утичу на стање порта, већ је *core* логика та која управља стањем на порту. У *Boundary Scan* режиму, *core* логика је изолована од порта а сигнали портова се контролишу преко *JTAG* интерфејса. *Boundary Scan* ћелије су повезане са серијским *shift* регистром који се користи за читање и уписивање стања на портовима [2].

2.3. *End-of-line* тестирање

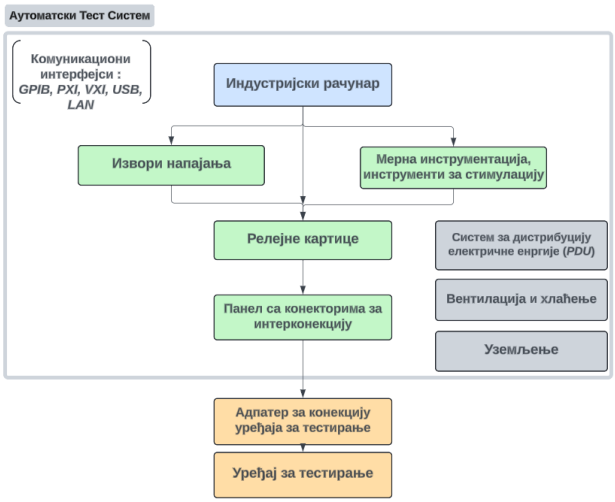
End-of-Line тестирање је кључна фаза у производном процесу, готови производи пролазе кроз свеобухватно тестирање како би се осигурало да испуњавају захтеве спецификација и стандарде квалитета пре него што буду испоручени купцима. Ова фаза тестирања проверава да ли производ правилно функционише, да ли је правилно састављен и да ли испуњава све перформансе и безбедносне критеријуме. Примарни циљ *End-of-Line* тестирања је потврда да је комплетан процес производње прошао без грешака и да сваки готов производ функционише по спецификацијама. Типичан *End-of-Line* тест укључује низ провера и мерења како би се потврдила функционалност и квалитет производа. Тачне компоненте *End-of-Line* процеса могу варирати у зависности од врсте производа, али неки од уобичајених елемената укључују: аутоматизована тест опрема, процедуре пре започињања фунцкионалног тестирања, функционално тестирање, тестирање перформанси и процедуре након завршетка тестирања [3].

3. СМЕРНИЦЕ ТОКОМ ДИЗАЈНА АУТОМАТИЗОВАНИХ ТЕСТ СИСТЕМА

Аутоматизована тест опрема или тест систем је сет подсистема који заједно функционишу како би тестирали одређени уређај или групу уређаја. Начин на који ови подсистеми комуницирају и међусобно се повезују значајно утиче на трошкове, перформансе, одржавање и употребљивост целокупног система. Из тог разлога, потребно је посветити време прецизном дефинисању архитектуре система од самог старта. На слици 1 приказана је архитектура генеричког аутоматизованог тест система.

Избор архитектуре тест система зависи од тога да ли је планирано да дизајнирани систем буде коришћен у производњи са великим обимом произведених уређаја, где је циљ да се тестови изврше за што краће време или у сврху валидације дизајна током развоја где је потребно прилагодити се честим променама дизајна а брзина извршавања теста није од суштинског значаја. Приликом избора архитектуре потребно је одабрати *rack-and stack* или *card-cage* инстуременте или одабрати хибридну архтектуру.

Rack-and-stack инструменти су самостални тестни уређаји који се могу користити независно. У тестним системима, често се слажу у орман (енг. *rack*, одакле и потиче назив) како би се уштедело на простору.



Слика 1. *Архитектура аутоматизованог тест система*

Са друге стране *card-cage* инструменти представљају интегрисане системе где се различите функционалности комбинују унутар заједничке картриџ структуре. Ипак, најчешћи избор је комбиновање оба типа инструмената у хибридну архитеткуру. Ова комбинација омогућује максималну флексибилност и функционалност, јер *card-cage* систем пружа интегрисану модуларну архитектуру, док *rack-and-stack* инструменти доприносе додатним могућностима кроз самосталне уређаје [4, 5].

3.1. Индустријски рачунар

За програмирање и контролу извршавања секвенци на аутоматизованом тест систему у већини случајева користи се индустријски рачунар који мора да задовољи критеријуме као што су велика процесна снага, различити типови комуникационих протокола као што су *GPIB, LAN, USB* итд. *GPIB* је дуго био стандардни интерфејс за повезивање мерних инструмената са рачунарима и омогућавање њихове програмабилне контроле. Иако *GPIB* остаје честа и корисна технологија, савремени рачунари често долазе са интегрисаним *LAN* и *USB* портовима, што је учинило и ове комуникационе протоколе индустријским стандардима [5,6].

3.2. Извори напајања

DC извор напајања је инструмент који служи да обезбеди једносмерни напон или струју уређају који тестирамо. Да би олакшали процесе тестирања, програмибилни извори напајања имају могућност повезивања са рачунаром преко различитих комуникационих протокола као и могућности мерења напона или струје на излазима. Два главна типа извора напајања су линеарна и прекидачка (енг. *switching*). Приликом избора напајања за тест систем, битно је узети у обзир конкретну примену. У зависности од примене, тестирани уређај може захтевати снагу од неколико миливата до више киловата. При избору извора напајања за тест систем потребно је водити рачуна о следећим карактеристикама: број потребних излаза, време стабилизације (енг. *Rise time and settling time*), излазни шум, брзина програмирања, брзина одговора на промене оптерећења (енг. *Transient*

response), могућности компензације пада напона у ожичењу (енг. *remote sensing*), заштите од високог напона или струје, уграђене могућности мерења итд.

Време пораста и време стабилизације су кључни показатељи способности напајања да достигне жељени ниво напона и стабилизује се. Конкретно, време пораста је време потребно да излаз напајања пређе са 10% на 90% подешене излазне вредности. Време стабилизације се описује као време потребно да се излазни напон стабилизује у оквиру одређеног процента своје коначне вредности, укључујући и време пораста.

Обзиром да проводници нису савршени и да у пракси чак и проводници направљени од материјала као што је бакар имају одређени отпор, у тест системима где се користи велики број жица за конекцију инструмената и уређаја који се тестирају долази до пада напона кроз жице, нарочито ако су у питању велике струје. На тај начин долази се у ситуације да је стварни напон на улазима уређаја за тестирање мањи од напона програмираног на извору напајања. За елиминацију ефекта пада напона на проводницима, потребно је користити методу познату као четворожичну (или Келвинову) технику. Основни принцип ове технике је коришћење одвојеног пара проводника за мерење напона на тестном уређају уз услов да кроз те проводнике не протиче струја.

Електрично оптерећење је програмабилни инструмент који кориснику нуди различите режиме рада као што су: константни напон, константа струја, константни отпор или константна снага. Електрично оптерећење је ефикасно решење за тестирање уређаја уместо коришћења фиксних вредности отпорника. Фиксни отпорник отежава аутоматизацију и симулирање стварног окружења уређаја који тестирамо као и прилогађавање променама у захтевима тестирања. Уколико се у току циклуса рада тест система јави захтев за променом вредности отпорности оптерећења, морали би физички променит отпорник. Електрична оптерећења су доступна у различитим димензијама, пружају велику флексибилност омогућавајући симулирање различитих нивоа и профила оптерећења и смањују трошкове тестирања [7].

3.3. Мерна инструментација

Мерни инструменти и инструменти за стимулацију који се користе у аутоматским тест системима одређени су функционалним и параметрисјким тестовима које је потребно извршити у току валидације уређаја који тестирамо. Потребно је анализирати све улазе и излазе на уређају, одредити тип тестова и мерења која су потребна за сваки појединачно и забележити потребну тачност и резолуцију мерења. Најчешћи мерни и стимулативни инструменти који се користе у аутоматским тест системима јесу дигитални мултиметри, осцилоскопи, сигнал генератори и *Data Aquisition* картице. Мерна и стимулативна инструментација се углавном бира у виду *cardcage* инструмената базираних на *PXI* архитектури. Главни разлог за одабир *PXI* iинструмената је модуларност и скалабилност [5,6].

3.4. Релејни систем

Прекидачи или релеји који међусобно повезују инструменте система и оптерећења са уређајем који се тестира представљају саставни део већине аутоматизованих тест система. Одабир одговарајуће врсте и топологије прекидача знатно утиче на трошкове, брзину, дуговечност и укупну функционалност аутоматског тест система. Основни типови релеја који се користе у изградњи аутоматских тест система су електромеханички релеји, *reed* релеји и *solid state* релеји.

Топологије релеја могу се поделити у три категорије: једноставне релејне конфигурације (енг. *General purpose relays*), мултиплексери и матрице. Одабир топологије за дизајн тест система зависи од броја инструмената који се користе, броја тест тачака на уређају, броја потребних истовремених конекција и потребне брзине тестирања. Организовање релеја у мултиплексер конфигурацију омогућује да се један улаз повеже на више излаза или обрнуто. Ова конфигурација је ефикасна уколико је потребно омогућити повезивање једног инструмента на више пинова уређаја који се тестира. Матрица има колоне и редове са релејем на сваком пресеку, што омогућује да се повежу сигнали између колона, колоне и реда, као и између редова [7].

Најчешћа архитетура тест система се базира на коришћењу комерцијално доступних релејних решења и назива се још централизована релејна архитектура. Ови инструменти углавном долазе у облику *card-cage VXI* или *PXI* архитетктуре и постављају се у орман тест система. На овај начин се знатно штеди време потребно за развој система и добија се на флексибилности. Мана овог приступа је повећана количина ожичавања тест система, а велика количина каблова може да изазове проблеме код осетљивих мерења.

Друга опција, уколико је потребно смањити количину жица унутар тест система и план за тестирање захтева већу прецизност мерења јесте да се систем релеја изведе унутар тест адаптера. Сигнали са мерних инструмената преусмеравају се на тест пинове уређаја за тестирање помоћу релеја постављених на штампану плочу постављену унутар тест адаптера. За реализацију овог типа повезивања потребно је додатно обезебедити могућност управљања релејима [6,8].

3.5. Систем за дистрибуцију електричне енергије

Инфраструктура напајања аутоматизованог тест система веома се разликује од уобичајених конфигурација напајања. Обзиром да се тестни системи састоје од више компоненти, захтевају сложеније системе напајања како би се осигурало да сваки део система добије потребну енергију. Тест системи из тог разлога користе јединице за расподелу напајања (енг. *Power Distribution Unit, PDU*) како би обезбедили да сва опрема у систему добија одговарајуће напајање. У току дизајна тест система потребно је израчуанти укупну струју појединачних инструмената и уверити се да главно напајање испуњава те захтеве. Уколико једнофазна линија

напајања не испуњаа захтеве, потребно је прећи на трофазно. У случају коришћења три фазе, неопходно је уверити се да инструменти у систему равномерно деле напајање са све три фазе [5,6].

Обзиром да се у току тестирања може десити да дође до губитка струје, пада напона или других кварова, заједно са јединицом за расподелу напајања у тест системима користе се извори непрекидног напајања. Њихова улога је осигурање да критичне компоненте у систему добију непрекидну енергију када дође до проблема са напајањем. У аутоматским тест системима углавном се користе два типа непрекидних напајања *Line interactive UPS* и *Double Conversion UPS.* Извори непрекидног напајања базирани на принципу рада *double conversion* методе представљају бољи избор за аутоматизовану тест опрему зато што у случају било какве грешке, време одзива уређаја је суштински непостојеће и напон на излазу је стабилан у случају губитка напона на главној линији [9].

3.6. Панел за интерконекцију

Панел за интерконекцију је кључни елемент савременог аутоматизованог тест система и представља механички носач дизајниран да олакша повезивање великог броја сигнала на тест систему. Помоћу панела за интерконекцију стандардни тест системи имају могућност да тестирају више различитих уређаја. Главне компоненте од којих се састоји панел за интерконекцију је стандардни интерфејс адаптер (енг. *Interface connector adapter*, *ICA*), више интерфејсних тест адаптера (енг. *ITA*) који омогућавају брзу и једноставну адаптацију тест система различитим уређајима за тестирање и конектора.

ICA је уобичајено монтирана на предњој страни тест система, на тест орману, и служи као оквир на који се позиционирају конектори са којима се врши ожичавање тест система са ресурсима као што су релејне картице, мерна инструментација и извори напајања. ITA је оквир на који се монтирају конектори који врше конекцију са уређајем за тестирање. Уобичајено се купује више ITA оквира, по један за сваки тип уређаја који се тестира [6, 10].

Virginia Panel Corporation (VPC) и *MAC Panel* су компаније специјализоване за производњу система за повезивање и интерконекцију у тестним окружењима, са посебним фокусом на висококвалитетна и поуздана решења за производне и лабораторијске примене

7. ЗАКЉУЧАК

Аутоматизација тестирања представља кључни фактор у осигурању високог квалитета и поузданости финалних производа, нарочито обзиром на све већу сложеност савремене електронике и растуће захтеве тржишта. Технике тестирања које се примењују на плочама пре њихове финалне монтаже омогућују рано откривање потенцијалних грешака у производном процесу, чиме се значајно смањују губици, који су тим већи што се грешке касније уоче.

У раду је посебно истакнут значај правилног дизајна аутоматизованих тест система, укључујући избор адекватних инструмената, рачунарске опреме и комуникационих протокола, што обезбеђује флексибилност и ефикасност производног процеса. Да би се обезбедила флексибилност потребно је детаљно анализирати захтеве које тест систем треба да испуни и правилно одабрати опрему која задовољава критеријуме тест плана али исто тако и предвидети могућности добијања нових захтева или редизајнирања тест система за нове производе. У погледу ефикасности, потребно је на основу обима производње и захтева за прецизност мерења одабрати инструменте који најефикасније извршавају дефинисане задатке.

8. ЛИТЕРАТУРА

[1] Charles Gallagher, Therese Lawlor-Wright, “A review of PCB design for In-Circuit testability guidelines and systems”, *Journal of Electronics Manufacturing, Vol. 5,No.3(September, 1995) 175-181*

[2] “Boundary Scan User’s Guide”, *Lauterbach Development Tools Release 02.2024.*

[3] “What is End-of-Line Testing In Manufacturing?”, Encida, https://encida.com/blog/what-is-end-of-line-testing (29.09.2024.)

[4] H. Alper Toku, “Developing New Automatic Test Equipments (ATE) using Systematic Design Approaches”, *Test Engineering Dept., Defence Systems Technologies Div.*

[5] “Test-System Development Guide- A comprehensive Handbook for Test Engineers”, *Agilent Technologies*

[6] National Instruments, “Designing Automated Test Systems - A Practical Guide to Software-Defined Test Engineering”

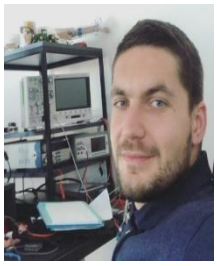
[7] *Keysight Techonoliges* , “The Power Handbook, A Guide for Test and Measurement Power Applications 2nd Edition”

[8] R. Hooper, "Optimal switching architecture for Automated Test Equipment" *2011 IEEE AUTOTESTCON, Baltimore, MD, USA, 2011, pp. 423-427, doi: 10.1109/AUTEST.2011.6058761*

[9] M. S. Racine, J. D. Parham and M. H. Rashid, "An overview of uninterruptible power supplies", *Proceedings of the 37th Annual North American Power Symposium, 2005., Ames, IA, USA, 2005, pp. 159-164, doi: 10.1109/NAPS.2005.1560518*

[10] “Mass interconnect for VXIbus Systems”, *https://www.artisantg.com/info/Keysight_Agilent_E3730A_Datasheet_20224893739.pdf?srsltid=AfmBOorGNplbQ0ZKY1W5NrylbuOFhySyPpNRd8YiwL80xqgk4fDhAfjz*

Кратка биографија:



Коста Босанчић рођен је у Кикинди 1994. године. Од 2019. године ради као инжењер аутоматске тест опреме у аутомобилској индустрији. Мастер рад на Факултету техничких наука из области Мерни Системи – Дизајн аутоматских тест система за функционално тестирање у серијској производњи одбранио је 2024. године. контакт: kosta_994@hotmail.rs

Зборник радова Факултета техничких наука, Нови Сад

УДК: (Upisuje redakcija, ostaviti ovaj red)
ДОИ: (Upisuje redakcija, ostaviti ovaj red)

ИМПЛЕМЕНТАЦИЈА СИСТЕМА ЗА БЕЛЕЖЕЊЕ GPS ДОГАЂАЈА УПОТРЕБОМ GEOFENCING ТЕХНОЛОГИЈЕ

IMPLEMENTATION OF A GPS EVENT TRACKING SYSTEM UTILIZING GEOFENCING TECHNOLOGY

Вук Вуковић, др Милан Видаковић, *Факултет техничких наука, Нови Сад*

Област – СОФТВЕРСКО ИНЖЕЊЕРСТВО И ИНФОРМАЦИОНЕ ТЕХНОЛОГИЈЕ

Кратак садржај – *Овај рад описује развој софтверског система за праћење GPS догађаја коришћењем geofencing технологије, која омогућава детекцију улазака и излазака корисника из дефинисаних зона, а ослања се на коришћење хаверсинусне формуле. Описани систем развијен је употребом Spring Boot и Flutter радних оквира.*

Кључне речи: *geofencing, хаверсинусна формула*

Abstract – *This paper outlines the creation of a GPS event tracking software system that utilizes geofencing technology to detect user entries and exits from specified zones, employing the haversine formula for precise calculations. The system was built using the Spring Boot and Flutter frameworks.*

Keywords: *geofencing, haversine formula*

1. УВОД

Технологије геолокације (геопозиције) [1] играју кључну улогу у развоју апликација које се ослањају на праћење кретања корисника у реалном времену. Геолокација представља механизам за одређивање географске локације уређаја. Једна од најзначајнијих метода која омогућава реаговање на кретање корисника је geofencing [2] технологија. Ова технологија омогућава дефинисање виртуелних граница око одређених тачака или географских области, као и праћење корисника, односно реаговање на улазак, излазак или задржавање у тој области одређено време.

Geofencing има примену у бројним индустријама, од безбедности, преко логистике, па све до маркетинга. Једна од специфичних примена geofencing-а јесте праћење кретања и времена проведеног на одређеним подручјима која имају стриктно дефинисане границе и очекивана задржавања, као што су гранични прелази.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Милан Видаковић, ред. проф.

1.1. Проблем

Проблем са којим се суочавају многи путници је недостатак информација о времену које је потребно да се проведе на граничном прелазу. Путници често немају довољно јасну слику о времену које ће изгубити током чекања у реду.

1.2. Решење

Коришћењем geofencing технологије могуће је детектовати када возило стигне до граничног прелаза, колико времена проводи у чекању, и када је границу коначно прешло. Овакве информације могу да буду од великог значаја за планирање пута.

2. ОПИС РЕШЕЊА

Описани проблем решен је имплементацијом система који се састоји из серверске стране и мобилне апликације. Бекенд је имплементиран користећи Spring Boot [3], док је мобилна апликација развијена употребом Flutter-a [4] и Google Maps API-ja [5]. Посебна пажња посвећена је имплементацији geofencing механизма унутар мобилне апликације.

2.1. Google Maps API

Google Maps API даје могућност укључивања мапа и функција заснованих на географским информацијама у апликације. Он пружа све што је потребно за приказивање мапа, али и проналажење локација и приказивање важних тачака на мапи.

У контексту овог решења, Google Maps API игра кључну улогу у приказивању граничних прелаза, праћењу корисниковог кретања и детекцији доласка на граничне прелазе и проласка кроз њих.

2.2. Geofencing

Geofencing је технологија која омогућава дефинисање виртуелних граница, односно области око физичких локација, као и праћење и реаговање на кретање уређаја, приликом интеракције са тим областима. Свака дефинисана област се назива geofence. Ове области се могу дефинисати око било које локације, и најчешће су кружног облика које се дефинише радијусом, односно полупречником кружнице. Чест случај су и правоугаони облици, а неретко се јављају и виртуелне границе неправилних облика. Примери различито дефинисаних области се могу видети на слици 1.

У контексту овог решења, мобилна апликација детектује и реагује на зоне граничних прелаза, односно аутоматски реагује на пристизање у ред граничног прелаза, као и преласка саме границе, тако да корисник не мора да води рачуна о томе током свог путовања.

Слика 1: примери geofence површина

2.3. Хаверсинусна формула

Технологија geofencing-а се ослања на математичке концепте, односно на прорачуне удаљености између две тачке на површини Земље. За прецизно израчунавање удаљености користи се хаверсинусна формула, дата у изразу 1. На основу те удаљености се одређује да ли се уређај, односно корисник налази у дефинисаној зони. Ова формула припада области сферне тригонометрије и одређује раздаљину две тачке на основу њихових географских дужина и ширина.

$$d = 2r \cdot \arcsin\left(\sqrt{hav(\Delta\phi) + \cos(\phi 1) \cdot \cos(\phi 2) \cdot hav(\Delta\lambda)}\right)$$

Израз 1: хаверсинусна формула

Формула је дефинисана на следећи начин:

- d - раздаљина између две тачке,
- r - полупречник Земље (приближно 6371 km),
- ϕ1 и ϕ2 – географске дужине две тачке, изражене у радијанима,
- Δϕ - разлика географских ширина (ϕ2 - ϕ1),
- Δλ - разлика географских дужина (λ2 - λ1), и
- $hav(x) = sin^2(\frac{x}{2})$.

3. GEOFENCING ИМПЛЕМЕНТАЦИЈА

3.1. Локацијске услуге

Пре свега, неопходно је затражити дозволу за приступ локацијским услугама мобилног уређаја. Уређаји ове услуге не пружају без дозволе. Ова дозвола је неопходна како би апликација могла да оствари увид у локацију уређаја, односно да прати кретање путника.

Приступ локацијским услугама се разликује у зависности од система. На Android платформи, дозволе се дефинишу у AndroidManifest.xml фајлу. На iOS платформи, дозволе се дефинишу у Info.plist фајлу, у којем је неопходно навести и разлоге због којих апликација тражи приступ локацији. Конфигурација за Android системе се налази у листингу 1, док се конфигурација за iOS системе налази у листингу 2.

```
<uses-permission
android:name="android.permission.ACCESS_FINE_LO
CATION"/>
<uses-permission
android:name="android.permission.ACCESS_COARSE_
LOCATION"/>
```

Листинг 1: Android конфигурација локацијских услуга

```
<key>NSLocationWhenInUseUsageDescription</key>
<string>We need your location to provide
automatic crossing services</string>
<key>NSLocationAlwaysUsageDescription</key>
<string>We need your location to provide
automatic crossing services</string>
```

Листинг 2: iOS конфигурација локацијских услуга

3.2. Geofencing

Следећи корак је довлачење података о граничним прелазима, односно података о geofence зонама са серверске апликације. Сваки гранични прелаз има дефинисану позицију која ће се користити за мерење удаљености и детекцију интеракције са geofence зоном.

Процес geofencing-а почиње рачунањем раздаљина од зоне уласка, а потом и зоне изласка граничног прелаза, употребом хаверсинусне формуле. Након што се раздаљине израчунају, проверавају се услови за детекцију уласка. Уколико се корисник налази у близини од 150 метара до дефинисане зоне и претходно није детектован улазак, апликација је у стању спремности да отпочне прелазак.

Након што је детектована зона, мора да се испуни још један услов. Чека се да корисник стане на крај реда чекања, како би мерење времена било максимално прецизно. Уколико брзина падне испод 2 m/s, односно 7,2 km/h, услов је задовољен, и тада се иницијализује догађај преласка, и поставља време доласка на гранични прелаз.

Док се апликација налази у стању активног преласка, све време се чека на догађај детекције зоне изласка, односно на испуњење последњег услова за завршетак циклуса. Када се корисник нађе на удаљености испод 100 метара од зоне изласка, процес се завршава, упућује се захтев ка серверској апликацији и бележи се време преласка, као и укупно чекање за дати гранични прелаз. Тада се апликација коначно враћа у почетно стање, где спремно чека на наредну детекцију geofence зоне.

3.3. Time-series подаци

На серверској страни, подаци су временски организовани, употребом TimescaleDB [6] базе података. TimescaleDB је специјализована база података за рад временским подацима (time-series data), која се заснива на PostgreSQL-у [7]. Овакав тип базе података идеалан је у случајевима када се бележе и анализирају велике количине података које су временски одређене, или које се прикупљају на фиксан временски интервал, као што су мерења сензора у

тачно одређено време, прикупљање логова, бележење GPS догађаја у датом тренутку, итд. TimescaleDB има све могућности стандардне PostgreSQL базе, али је додатно оптимизована када су временски одређени подаци у питању.

TimescaleDB чува податке у тзв. хипертабеле (hypertables) [8], које се сегментишу на основу временских интервала. Такав приступ складиштењу података омогућава брже чување и претрагу података, што може бити изузетно корисно за системе који континуирано бележе и анализирају временски одређене догађаје.

4. ЗАКЉУЧАК

Кроз овај рад развијена је серверска апликација употребом Spring Boot радног оквира, уз TimescaleDB временску базу података. Поред тога, развијена је мултиплатформска мобилна апликација користећи Flutter framework, уз ослонац на Google Maps API.

Посебна пажња је посвећена истраживању и имплементацији geofencing технологије, која омогућава детекцију уласка и изласка уређаја, то јест корисника из дефинисаних географских подручја. Ова технологија, примењујући хаверсинусну формулу, представља срж система за аутоматско праћење прелазака граница. Ове компоненте и технологије заједно чине функционалан и аутоматизован систем који корисницима нуди тачне и правовремене информације о стању на граничним прелазима.

5. ЛИТЕРАТУРА

[1] https://www.indicative.com/resource/geolocation (приступљено у октобру 2024.)
[2] Dmitry Namiot, Manfred Sneps-Sneppe, “Geofence and Network Proximity“, Lecture Notes in Computer Science, vol 8121. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-40316-3_11
[3] https://spring.io/projects/spring-boot (приступљено у октобру 2024.)
[4] https://flutter.dev (приступљено у октобру 2024.)
[5] https://developers.google.com/maps (приступљено у октобру 2024.)
[6] https://www.timescale.com(приступљено у октобру 2024.)
[7] https://www.postgresql.org (приступљено у октобру 2024.)
[8] https://docs.timescale.com/use-timescale/latest/hypertables (приступљено у октобру 2024.)

Кратка биографија:

Вук Вуковић рођен је 2000. године у Београду. Академске студије завршио је на Рачунарском факултету у Београду, на смеру рачунарске науке, након чега је уписао мастер академске студије на Факултету техничких наука у Новом Саду, на студијском програму софтверско инжењерство и информационе технологије.
Контакт имејл: vuk.vukovic.2000@gmail.com
Контакт мобилни: +381 62 230056

Milan Vidaković rođen je u Novom Sadu 1971. Doktorirao je na Fakultetu tehničkih nauka 2003. god, a od 2014. je u zvanju redovni profesor iz oblasti Primenjene računarske nauke i informatika.

UDK: 4.42
DOI: https://doi.org/10.24867/31BE30Vukovic

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.9
DOI: https://doi.org/10.24867/31BE31Poznanovic

DOKAZI NULTOG ZNANJA

ZERO KNOWLEDGE PROOFS

Isidora Poznanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – *U ovom radu je predstavljen koncept dokaza nultog znanja, njihov kriptografski značaj i primene. Prikazana je osnovna podela na interaktivne i neinteraktivne dokaze nultog znanja. Prikazana su i upoređena tri neinteraktivna protokola nultog znanja: ZK-SNARK, ZK-STARK i Bulletproofs. Predstavljen je problem kvadratnog ostatka koji je dokazan pomoću oba tipa dokaza nultog znanja. Neinteraktivni protokol koji je korišćen za dokazivanje problema kvadratnog ostatka je ZK-SNARK. Dokaz je realizovan u programskom jeziku Python, uz pomoć python-snark biblioteke.*

Ključne reči: *Dokaz nultog znanja, ZK-SNARK, ZK-STARK, Kriptografija*

Abstract – *This paper presents zero knowledge proofs, their cryptographic significance and applications. It presents a basic classification: interactive and non-interactive zero knowledge proofs. It presents and compares three protocols of non-interactive zero knowledge proofs: ZK-SNARK, ZK-STARK and Bulletproofs. It presents the quadratic residue problem and proofs it with both interactive and non-interactive zero knowledge proofs. The non-interactive protocol used to prove the quadratic residue problem is ZK-SNARK. The proof is implemented in the Python programming language, using python-snark library.*

Keywords: *Zero knowledge proof, ZK-SNARK, ZK-STARK, Cryptography*

1. UVOD

Pojam dokaza je opšte poznat i široko korišćen u raznim oblastima različitih nauka. Najtipičniji primer dokaza je matematički dokaz. Matematički dokaz se u osnovi sastoji iz fiksnog broja koraka pomoću kojih se iz opšte prihvaćenih činjenica (aksioma) dolazi do tvrdnje koja je dokazivana. Dokazi nultog znanja imaju isti cilj kao i matematički dokazi, da validiraju ispravnost neke tvrdnje, međutim način na koji to rade je drugačiji.

Inicijalno, dokazi nultog znanja su bili interaktivni, pa su bili sličniji dokazima u zakonu ili debati nego matematičkim dokazima, jer se istinitost tvrdnje u ovakvim dokazima verifikuje dinamičkom interakcijom više strana.

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

Interakcija može biti skupa i spora, pa su uloženi napori da se ona vremenom izbaci iz dokaza nultog znanja, koji na taj način postaju neinteraktivni. Neinteraktivni dokazi mogu se porediti sa dokazima iz matematike ili formalne logike. Analogno dokazima u naučnim radovima, neinteraktivni dokazi nultog znanja trebaju biti univerzalni i sadržati sve potrebne informacije za validaciju bez ikakve dodatne interakcije. Ali za razliku od dokaza u naučnim radovima, neinteraktivni dokazi nultog znanju ne smeju otkrivati nikakve dodatne informacije osim činjenice da je dokazivana tvrdnja tačna.

2. DEFINICIJA DOKAZA NULTOG ZNANJA

Dokazi nultog znanja su se prvi put pojavili 1985. godine, u radu *The knowledge complexity of interactive proof systems* koji su objavili *Shafi Goldwasser*, *Silvio Micali* i *Charles Rackoff*. U tom radu je data definicija dokaza nultog znanja koja se i danas koristi [1]:

Protokol nultog znanja je metod kojim jedna strana (dokazivač) može dokazati drugoj strani (verifikatoru) da je određena tvrdnja tačna, bez otkrivanja bilo kakvih informacija osim činjenice da je određena tvrdnja tačna.

Navedeni rad daje matematičke osnove ovog metoda i navodi prve primere dokaza nultog znanja. Dokazi nultog znanja su se od tada konstantno unapređivali, pronašli su upotrebu u realnosti i danas su široko korišćeni. Zahvaljujući ovom radu postavljena je osnova za mnoge kriptografske protokole koji se i danas koriste.

U nastavku je dat primer u kome se mogu koristiti dokazi nultog znanja kao i benefiti njihovog korišćenja:

Postoji tvrdnja *"Ja imam više od 18 godina"* koju korisnik želi da dokaže nekom servisu. Kako bi korisnik to dokazao potrebno je da dostavi važeći identifikacioni dokument, poput lične karte ili pasoša, koji sadrži njegovu godinu rođenja. Dostavljanjem ličnog dokumenta servisu korisnik se izlaže riziku, rizikuje svoju privatnost. Najveća opasnost koja vreba korisnika je krađa identiteta. Ona se najčešće dešava usled hakerskih napada kada servisi skladište podatke u centralizovanim bazama podataka.

U ovakvim situacijama bolje je koristiti dokaze nultog znanja koji nisu podložni rizicima standardnog pristupa. Protokol nultog znanja koristi, kao ulaz, početnu tvrdnju na osnovu koje generiše sažet dokaz njene istinitosti. Dokaz pruža garanciju da je tvrdnja istinita bez upotrebe informacije koja je korišćena da se on napravi. Stoga kako bi korisnik dokazao servisu da ima više od 18 godina potrebno je da priloži dokaz nultog znanja. Servis treba da proveri da su odgovarajuća svojstva dokaza tačna i znaće

da je dokazivana tvrdnja tačna. U nastavku će biti razmatrano koja su to svojstva, kakvi sve nulti dokazi postoje i kako funkcionišu.

Slika 1. *Vizuelni opis dokaza nultog znanja [2]*

3. ELEMENTI DOKAZA NULTOG ZNANJA

U dokazima nultog znanja učestvuju dve strane: dokazivač i verifikator. Ukoliko se dokazivač ili verifikator pridržavaju protokola nultog znanja korektno, kažemo da su pošteni. Svaki dokaz nultog znanja neke tvrdnje mora zadovoljavati sledeća tri svojstva:

1. **Kompletnost** - ukoliko je tvrdnja tačna, protokol sa nultim znanjem uvek vraća *tačno*. Bilo koji pošteni dokazivač će ubediti bilo kog poštenog verifikatora u istinitost tvrdnje.
2. **Ispravnost** - ukoliko je tvrdnja netačna, teorijski je nemoguće prevariti protokol nultog znanja da vrati *tačno*. Ni jedan lažljivi dokazivač ne može prevariti ni jednog poštenog verifikatora da poveruje da je netačna tvrdnja tačna (postoji izuzetak zanemarivo male verovatnoće).
3. **Nulto znanje** - verifikator ne može naučiti ništa o tvrdnji osim njene istinitosti.

4. TIPOVI DOKAZA NULTOG ZNANJA

Dokazi nultog znanja mogu biti interaktivni ili neinteraktivni. Interaktivni dokazi uključuju uzajamnu komunikaciju između dokazivača i verifikatora, dok neinteraktivni dozvoljavaju dokazivaču da generiše jedan dokaz koji će verifikator kasnije koristiti nezavisno od dokazivača.

4.1. Interaktivni dokazi nultog znanja

U svojoj osnovnoj formi dokazi nultog znanja su interaktivni i sastoje se iz tri elementa: svedoka, izazova i odgovora.

- Svedok - Dokazivač želi da dokaže znanje o nekoj skrivenoj informaciji verifikatoru. Ta skrivena informacija je *svedok* dokaza. Dokazivač na osnovu znanja o svedoku pravi grupu pitanja na koje može odgovoriti samo strana koja ima znanje o skrivenoj informaciji. Dakle, dokazivač započinje proces dokazivanja tako što nasumično bira pitanje, računa odgovor i šalje ga verifikatoru.
- Izazov - Verifikator nasumično bira drugo pitanje i pita dokazivača da mu odgovori.
- Odgovor - Dokazivač prihvata pitanje, računa odgovor i vraća ga verifikatoru. Na osnovu odgovora dokazivača, verifikator može otkriti da li on ima skrivenu informaciju. Da bi se osigurao da dokazivač ne nagađa odgovore, verifikator može slati više pitanja. Pitanja šalje sve dok ne bude smatrao da je verovatnoća da dokazivač slučajno pogađa tačne odgovore dovoljno mala.

4.2. Neinteraktivni dokazi nultog znanja

Iako su interaktivni dokazi nultog znanja bili značajno otkriće, potreba za konstantnom povezanošću i interakcijom između dve strane je značajno ograničavala njihovu primenu. Čak i kada bi verifikator bio ubeđen u znanje dokazivača dokaz bi bio nedostupan nekom drugom verifikatoru koji želi da proveri istu informaciju. Kako bi se ovaj problem rešio predstavljeni su neinteraktivni dokazi nultog znanja u kojima verifikator i dokazivač imaju deljeni ključ [3].

Za razliku od interaktivnih dokaza, neinteraktivni zahtevaju jednokratnu komunikaciju između verifikatora i dokazivača. Dokazivač prosleđuje tajnu informaciju specijalnom algoritmu koji računa dokaz nultog znanja. Dokaz se šalje verifikatoru koji proverava da li dokazivač zna tajnu informaciju koristeći drugi algoritam. Jednom kada je dokaz izgenerisan on je dostupan i drugim verifikatorima koji imaju pristup deljenom ključu i verifikacionom algoritmu. Pritom verifikator nije u stanju da rekonstruiše originalnu informaciju iz dokaza.

Ovaj tip dokaza je posebno koristan u slučaju da je komunikacija između strana ograničena ili skupa. Neinteraktivni dokazi nultog znanja imaju dosta podtipova koji su prevashodno razvijeni za različite potrebe primene.

4.2.1. ZK-SNARK

ZK-SNARK (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge) je protokol nultog znanja koji pravi sažeti dokaz koji može biti proveren brzo i ne iziskuje interaktivnu komunikaciju dokazivača i verifikatora [4].

Ovim protokolom su zadovoljena ranije pominjana svojstva kompletnosti, ispravnosti i nultog znanja. Sažetost dokaza generisanih uz pomoć ZK-SNARK protokola se ogleda u maloj veličini dokaza i efikasnoj verifikaciji.

Veličina generisanih dokaza je konstantna ili raste logaritamski u odnosu na veličinu aritmetičkog kola kojim se ispituje tačnost zadate tvrdnje. Pa je veličina dokaza kompleksnih problema gotovo jednaka veličini dokaza jednostavnih problema i obično iznosi par stotina bajtova.

Verifikacija dokaza je kratka i izvršava se u mnogo kraćem vremenskom periodu nego računanja u aritmetičkom kolu [5].

4.2.2. ZK-STARK

ZK-STARK (Zero-Knowledge Scalable Transparent Argument of Knowledge) je protokol predstavljen u radu [6] iz 2018 godine. Protokol je sličan prethodno objašnjenom protokolu, najbitnije razlike se ogledaju u pojmovima skalabilnosti i transparentnosti.

Skalabilnost je jedna od ključnih osobina ovog protokola, ona omogućava način da se kreira sažet i efikasan dokaz za širok spektar proračuna. Pa je ovaj protokol pogodan za različite aplikacije uključujući blokčejn i decentralizovane sisteme. ZK-STARK je dosta brži u generisanju i verifikovanju dokaza kako veličina svedoka raste, sa povećanjem svedoka blago se linearno povećavaju vremena da se izgeneriše i verifikuje dokaz.

ZK-STARK je dizajniran da bude transparentan. Njegova sigurnost se ne oslanja na nedokazane pretpostavke i kompleksne matematičke strukture. Nasumičnost koja se koristi za generisanje parametara za dokazivanje i verifikaciju je javno dostupna te nema potrebe za pouzdanom ceremonijom postavljanja ključeva.

4.2.3. Bulletproofs

Bulletproofs protokol je prvi put predstavljen u radu [7] iz 2018. godine kao protokol sa neprobojnim sigurnosnim pretpostavkama, kratak kao metak. Dizajniran je tako da u određenim scenarijima bude skalabilniji i efikasniji od ZK-SNARK protokola. Najbolje se primenjuje u dokazima opsega, gde dokazivač želi da pokaže da se tajna vrednost nalazi unutar nekog opsega bez otkrivanja ostalih informacija o tajnoj vrednosti. Ovaj protokol nultog znanja generiše kratke dokaze logaritamske veličine u odnosu na veličinu svedoka i nije mu potrebna pouzdana ceremonija postavke ključeva. Bulletproofs protokol se kao i ZK-SNARK oslanja na kriptografiju eliptičnih kriva, čija sigurnost koja je zasnovana na tome da je izuzetno teško izračunati diskretni logaritam u eliptičkim krivama. Za klasične računare, diskretni logaritamski problem na eliptičkim krivama je eksponencijalno težak ali je u kvantnim računarima ovaj problem rešiv u polinomijalnom vremenu.

Ovaj protokol je usvojen od strane raznih kriptovaluta poput Monera. Pri prelasku na Bulletproofs zapaženo je smanjenje od 80% u veličinama transakcija.

4.2.4. Poređenje neinteraktivnih protokola

U nastavku je prikazan odnos prethodno opisanih neinteraktivnih protokola. Najveća mana i slaba tačka ZK-SNARK protokola nalazi se u ceremoniji postavljanja ključeva. U protokolu ZK-STARK te ceremonije nema, jer on koristi transparentne heš protokole sa polinomkim IOP [8]. Bulletproofs protokol koristi argumente unutrašnjeg proizvoda zasnovane na eliptičkim krivama kojima nije potrebno pouzdano podešavanje ključeva.

ZK-SNARK ima konstantnu veličinu dokaza, dokazi su kratki i ne zavise od kompleksnosti problema. Nešto veći dokazi kreiraju se kroz Bulletproofs protokol, oni rastu logaritamski sa porastom kompleksnosti, dok ZK-STARK generiše najveće dokaze koji su polilogaritamski. Ali, bez obzira što su dokazi ovog protokola najduži, oni se generišu u kvazilinearnom vremenu pa se time i najbrže generišu, jer ostala dva protokola iziskuju linearno vreme. Kada je u pitanju vreme verifikacije dokaza, najmanje vremena je potrebno ZK-SNARK protokolu, a najviše Bulletproofs protokolu [9].

5. KRIPTOGRAFSKI ZNAČAJ

Matematika kriptografije je inicirana primenama u stvarnom svetu. Najosnovnija i prvobitna primena ogleda se u želji da se privatno komunicira u prisustvu prisluškivača koji osluškuje komunikaciju. Sa pojavom računara kao sredstva za komunikaciju, pojavljuju se brojne druge primene, od verifikacije autentičnosti podataka i pristupnih privilegija do omogućavanja složenih finansijskih transakcija preko interneta koje uključuju više strana, od kojih svaka ima svoje poverljive informacije.

5.1. Upotreba dokaza nultog znanja u kriptografiji

Goldreich, *Micali* i *Wigderson* su u radu *How to Prove All NP Statements in Zero-Knowledge* 1987. godine dokazali da se za svaki problem iz *NP* klase problema može konstruisati dokaz nultog znanja. Ova činjenica nameće dokaze nultog znanja kao veoma moćan alat u modernoj kriptografiji. Veza kriptografije i dokaza nultog znanja nalazi se u konceptu zaštite poverljivih informacija i osiguravanja bezbednosti podataka. Kriptografija omogućava alate za zaštitu informacija, dok dokazi nultog znanja pružaju sigurnu proveru tih informacija bez otkrivanja poverljivih podataka. Dokazi nultog znanja koriste se kao metoda za očuvanje sigurnosti i poverljivosti u teorijskim osnovama kriptografije, u zadacima poput generatora pseudo-nasumičnih brojeva i enkripcije. Na primer, dokazi nultog znanja omogućavaju dokazivanje autentičnosti podataka bez otkrivanja samih podataka u složenim finansijskim transakcijama u koje je uključeno više strana sa svojim sopstvenim poverljivim informacijama [10,11].

Dokazi nultog znanja su uveli jedinstven pristup u oblasti kriptografije i izdvojili su se od drugih kriptografskih protokola koji su usmereni na privatnost u distribuiranim sistemima. U master radu upoređeni su sa kriptografskim metodama homomorfnog šifrovanja i sigurnih višestranih proračuna (eng. *Secure Multiparty Computation*). Svi ovi metodi za svrhu imaju verifikaciju informacija i očuvanje privatnosti [12]. Homomorfno šifrovanje omogućava izvođenje proračuna nad šifrovanim podacima bez potrebe za dešifrovanjem [13]. Dok sigurnosni višestrani proračuni omogućavaju nepoverljivu saradnju više strana koje zajedno računaju zadatu funkciju preko njihovih ulaza koji ostaju privatni [14].

5.1. Zcash

Zcash je jedna od najznačajnih primena ZK-SNARK protokola zato što on ujedno predstavlja i prvu široko korišćenu primenu dokaza nultog znanja u praksi. Dokazima nultog znanja osigurava zaštićene transakcije u kojima pošiljalac, primalac i količina resursa prenetih transakcijom ostaju privatni [15,16].

Zcash je kriptovaluta koja je fokusirana na privatnosti i anonimnosti transakcija. Ugrubo nastao je tako što je grupa naučnika na Bitkoinov otvoreni kod dodala dokaze nultog znanja [17]. Motivacija za tim je nedostatak privatnosti Bitkoina gde su transakcije verifikovane i snimljene na javnom blokčejnu, pa svako može pristupiti korisničkim balansima i podacima o transakcijama.

Ova kriptovaluta implementira Decentralized Anonymous Payment (DAP) šemu pod nazivom Zerocash koja je detaljno opisana u radu iz 2014. godine [18]. Razlika između DAP šeme i standardnog Bitkoin sistema jeste u tome što Bitkoin sistem koristi transparentne transakcije, dok DAP šema omogućava korisnicima da vrše zaštićene transakcije (shielded transakcije). Ove transakcije skrivaju osetljive informacije, tj. skrivaju informacije o pošiljaocu i primaocu, kao i iznos transakcije. Međutim Zcash pored zaštićenih transakcija omogućava i transparentne transakcije slične onima koje Bitkoin koristi. Stoga korisnici mogu sami izabrati nivo privatnosti koji je njima potreban [19].

Provera validnosti transakcija bez otkrivanja podataka o transakciji omogućena je korišćenjem ZK-SNARK protokola. Pomoću ovog protokola učesnici u transakciji mogu da dokažu da imaju informacije koje su potrebne za izvršenje transakcije (na primer privatne ključeve), bez otkrivanja tih informacija trećoj strani. Zaštićene transakcije mogu biti u potpunosti enkriptovane u blokčejnu, ali se i dalje mogu verifikovati kao validne pod uslovima konsenzusa koristeći ZK-SNARK protokol [20].

6. ZAKLJUČAK

Dokazi nultog znanja predstavljaju revolucionarno otkriće u oblasti kriptografije, obezbeđujući sredstvo za dokazivanje znanja bez otkrivanja samog znanja. U ovom radu navedena je prva definicija ovog koncepta i osnovna podela na interaktivne i neinteraktivne dokaze nultog znanja. Upoređena su tri najpoznatija neinteraktivna dokaza nultog znanja: ZK-SNARK, ZK-STARK i Bulletproofs.

Kako tehnologija nastavlja da se razvija, primena dokaza nultog znanja se sve više širi. Dokazi nultog znanja su danas jedan od najpopularnijih metoda za osiguravanje anonimnosti transakcija pa igraju ključnu ulogu u razvoju bezbednih digitalnih sistema koji čuvaju privatnost korisnika.

7. LITERATURA

[1] Shafi Goldwasser, Silvio Micali, Sharles Rackof, "*The knowladge complexity of interactive proof systems*", Society for Industrial and Applied Mathematics, 1985.

[2] https://schor.medium.com/on-zero-knowledge-proofs-in-blockchains-14c48cfd1dd1 (pristupljeno u oktobru 2024.)

DOI: https://doi.org/10.24867/31BE31Poznanovic

[4] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, Madars Virza, *"Succinct Non-Interactive Zero Knowledge for a von Neumann Architecture"*, 23rd USENIX Security Symposium (USENIX Security 14). San Diego, CA: USENIX Association, Avg. 2014., str. 781-796. ISBN: 978-1-931971-15-7. Dostupno na: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/ben-sasson (pristupljeno u oktobru 2024.)

[5] Nir Bitansky, Ran Canetti, Alessandro Chiesa, Eran Trome, *"Recursive Composition and Bootstrapping for SNARKs and Proof-Carrying Data"*, 2012.

[6] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, Michael Riabzev, *" Scalable, transparent, and post-quantum secure computational integrity"*, 2018.

[7] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, Greg Maxwell, *"Bulletproofs: Short Proofs for Confidential Transactions and More"*, 2017.

[8] Szepieniec Alan, Zhang Yuncong, *"Polynomial IOPs for Linear Algebra Relations"*, Springer International Publishing, 2022., str. 523–552. ISBN: 978-3-030-97121-2.

[9] El-Hajj Mohammed, Oude Roelink Bjorn, *"Evaluating the Efficiency of zk-SNARK, zk-STARK, and Bulletproof in Real-World Scenarios: A Benchmark Study"*, 2024. Dostupno na: https://www.mdpi.com/2078-2489/15/8/463 (pristupljeno u oktobru 2024.)

[10] Shafi Goldwasser, *" Mathematical foundations of modern cryptography: computational complexity perspective"*, 2002. arXiv: cs / 0212055 (cs.CR). Dostupno na: https://arxiv.org/abs/cs/0212055 (pristupljeno u oktobru 2024.)

[11] Goldreich Oded, Micali Silvio, Wigderson Avi., *"How to Prove all NP Statements in Zero-Knowledge, and a Methodology of Cryptographic Protocol Design"*, sv. 263 1986., str. 171-185.

[12] Ryan Lavin, Xuekai Liu, Hardhik Mohanty, Logan Norman, Giovanni Zaarour, Bhaskar Krishnamachari "*A Survey on the Applications of Zero-Knowledge Proofs*", 2024. arXiv: 2408.00243 (cs.CR). Dostupno na: https://arxiv.org/abs/2408.00243 (pristupljeno u oktobru 2024.)

[13] Ronald L. Rivest and Michael L. Dertouzos, *"On data banks and privacy homomorphisms"*, 1978. Dostupno na: https://api.semanticscholar.org/CorpusID:6905087 (pristupljeno u oktobru 2024.)

[14] Yao, Andrew C., *"Protocols for secure computations"*, 23rd Annual Symposium on Foundations of Computer Science (sfcs 1982). 1982., str. 160-164.

[15] What are zero-knowledge proofs, Dostupno na: https://z.cash/learn/what-are-zero-knowledge-proofs/ (pristupljeno u oktobru 2024.)

DOI: https://doi.org/10.24867/31BE31Poznanovic

[17] How is Zcash different than Bitcoin, Dostupno na: : https://z.cash/learn/how-is-zcash-different-than-bitcoin/ (pristupljeno u oktobru 2024.)

DOI: https://doi.org/10.24867/31BE31Poznanovic

[19] Sean Bowe, Daira Hopwood, Taylor Hornby, Nathan Wilcox, *"Zcash Protocol Specification"*, 2020.

[20] What are ZK-SNARKS, Dostupno na: https://z.cash/learn/what-are-zk-snarks/ (pristupljeno u oktobru 2024.)

Kratka biografija:

Isidora Poznanović rođena je u Kragujevcu 2000. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehničko i računarsko inženjerstvo – računarstvo i automatika, odbranila je 2024.god.

Kontakt: isidora@uns.ac.rs

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.738.5:339
DOI: https://doi.org/10.24867/31BE32Fruza

PROJEKTOVANJE OSNOVNE INFRASTRUKTURE ZA HOSTOVANJE VEB APLIKACIJA

DESIGNING OF THE BASIC INFRASTRUCTURE FOR HOSTING WEB APPLICATIONS

Denis Fruža, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu je prikazan proces dizajniranja i implementacije osnovne *AWS cloud* infrastrukture za hostovanje veb aplikacija uz korišćenje *Terraform IaC (Infrastructure as code) alata. Akcenat je stavljen na postavljanje visoko dostupne, skalab*ilne i bezbedne infrastrukture. Takođe su opisane i najbolje prakse za projektovanje *VPC*-a, upravljanje resursima kao i upotreba nekih od bitnih *AWS* servisa.

Ključne reči: *cloud tehnologije, infrastruktura, internet mreže, AWS, Terraform, veb aplikacija*

Abstract – *This paper presents the process of designing and implementing the basic AWS cloud infrastructure for hosting web applications using Terraform IaC (Infrastructure as Code) tools. The focus is on setting up a highly available, scalable, and secure infrastructure. Best practices for VPC design, resource management, as well as the use of some essential AWS services, are also described.*

Keywords: *cloud technologijes, infrastructure, internet networks, AWS, Terraform, web application*

1. UVOD

U trenutnoj fazi digitalne revolucije, *cloud computing* se istakao kao nezaobilazna komponenta svake *IT* infrastrukture. On omogućava organizacijama da lako upravljaju svojim resursima, optimizuju troškove i unaprede dostupnost svojih aplikacija. *Amazon Web Services* (*AWS*) se nalazi na čelu ove industrije i svojim korisnicima nudi preko 200 različitih servisa iz globalno raspoređenih *data* centara.

Iako *cloud* servisi pojednostavljuju procese, upravljanje složenijim sistemima može biti veoma izazovno što naglašava značaj upravljanja infrastrukturom uz pomoć *infrastructure as code* (*IaC*) pristupa. Terraform, kao vodeći alat u ovoj oblasti, omogućava automatizaciju i efikasno upravljanje *cloud* infrastrukturom kroz kod.

Na početku rada biće prikazane osnove cloud computinga, ključni *AWS* servisi i najbolje prakse za njihovu upotrebu, s posebnim naglaskom na *Virtual Private Cloud* (*VPC)* i mrežnu bezbednost kao osnovu za kreiranje visoko dostupne, skalabilne i bezbedne infrastrukture. Takođe će biti objašnjen i *IaC* koncept i korišćenje Terraform-a za automatizaciju i upravljanje infrastrukturom. Nakon potrebnog predznanja, detaljno će biti prikazan proces dizajniranja i implementacije infrastrukture za razvoj i skaliranje veb aplikacija.

2. TEORIJSKE OSNOVE

Cloud computing predstavlja skup tehnologija koje se koriste za isporuku računarskih resursa kojima mogu pristupiti korisnici širom sveta. Osnovna ideja je omogućiti pristup podacima bez obzira na trenutno okruženje korisnika. Krajnji korisnici pristupaju *cloud* aplikacijama putem veb, mobilnih ili desktop aplikacija. Postoje tri tipa servisa u *cloud computing*-u:

- infrastruktura kao servis (IaaS) - *cloud* provajder upravlja fizičkom infrastrukturom, a klijent ima mogućnost da na toj infrastrukturi upravlja operativnim sistemom, mrežom i skladištem. Nudi najširi spektar mogućnosti i pristupa.
- platforma kao servis (PaaS) - *cloud* provajder klijentima dodatno pruža i radni okvir (*framework*), aplikativne programske interfejse (*API*) i druge alate neophodne za razvoj, isporuku i testiranje aplikacija. Klijent upravlja svojim aplikacijama koje instalira na datoj infrastrukturi i u određenoj meri njihovom konfiguracijom.
- softver kao servis (SaaS) - klijentima (korisnicima aplikacija) pruža se gotov softver na korišćenje. Provajder tog softvera u potpunosti upravlja infrastrukturom i aplikacijama na infrastrukturi.

Neki od popularnih *cloud* servis provajdera su *Amazon Web Services* (*AWS*), *Microsoft Azure*, *Google Cloud Platform* (*GCP*) i *Digital Ocean*.

Ovaj rad proistekao je iz master rada čiji mentor je bio doc. dr Željko Vuković

2.1. *AMAZON WEB SERVICES* (*AWS*)

Amazon Web Services (*AWS*) predstavlja *cloud* platformu veb servisa koja pruža kompletna rešenja u mnogim domenima računarstva kao što su skladištenje i održavanje baza podataka, obrada samih podataka, pružanje infrastrukture za najkompleksnija rešenja, mašinsko učenje i drugo. Ukupan broj servisa koje pruža je preko 200 i u 2022. godini zauzimao je 34% čitavog *cloud* tržišta.

2.2. INFRASTRUKTURA KAO KOD

Infrastruktura kao kod (*IaC*) predstavlja moderan pristup pri upravljanju infrastrukturom. Ovaj koncept transformiše manuelni u automatizovani pristup upravljanja infrastrukturom uz pomoć koda. Kompletna konfiguracija i postavka infrastrukture opisana je u lako čitljivim konfiguracionim fajlovima. *IaC* u velikoj meri pojednostavljuje procese implementacije, održavanja i skaliranja infrastrukture. Najpopularniji *IaC* alati su *Terraform*, *Puppet*, *AWS CloudFormation*, Ansible i drugi.

3. *AWS* I NJEGOVI SERVISI

Neki od najpopularnijih servisa na *AWS*-u kao što su *Virtual Private Cloud* (*VPC*), Elastic *Compute Cloud* (*EC2*), *Relational Database Service* (*RDS*), *Elastic Load Balancing* i *Autoscaling, Simple Storage Service* (*S3*) kao i drugi servisi korišćeni su u projektovanju i implementaciji infrastrukture u ovom master radu. U daljem tekstu biće opisani neki od korišćenih servisa:

- *Virtual Private Cloud* (*VPC*) - predstavlja logički izolovanu privatnu mrežu u kojoj se postavljaju resursi. Dozvoljava kreiranje podmreža, konfigurisanje IP adresa, bezbednosnih zaštita, rutiranja saobraćaja i omogućava potpunu kontrolu nad mrežom.
- *Elastic Compute Cloud* (*EC2*) - servis koji omogućava zauzimanje virtuelnih mašina (*EC2* instanci) za potrebe hostovanja aplikacija, skladištenja, obrade i analize podataka itd.
- *Relational Database Service* (*RDS*) - servis za upravljanje relacionim bazama podataka. Omogućava veoma jednostavno postavljanje, upravljanje i skaliranje relacionih baza podataka kao što su *MySQL*, *PostgreSQL*, *Amazon Aurora*, *MariaDB* itd. Sve bitne operacije poput pravljenja rezervnih kopija (*backup*-a), replikacija, oporavaka od grešaka ili vraćanja baze podataka obavljaju se automatski od strane *AWS*-a.
- *Elastic Load Balancing* i *Autoscaling* – predstavljaju servise koji imaju ulogu u postizanju maksimalne dostupnosti i skalabilnosti infrastrukture. *Load Balancer*-i vrše distribuciju saobraćaja na više različitih ciljeva unutar jedne ili više zona dostupnosti. *Auto Scaling* omogućava automatsko prilagođavanje broja resursa (npr. *EC2* instanci) na osnovu unapred definisanih uslova i metrika i ima veoma važnu ulogu kako u skaliranju infrastrukture, tako i u optimizaciji troškova.
- *Simple Storage Service* (*S3*) - predstavlja skalabilan, visoko dostupan i siguran servis za skladištenje podataka. Namenjen je za skladištenje i dobavljanje velike količine podataka po veoma niskoj ceni. Nema nikakve limite kada su u pitanju količina skladištenih podataka kao i broj konkurentnih preuzimanja istih.

4. *TERRAFORM* - INFRASTRUKTURA KAO KOD

Terraform je alat otvorenog koda koji se koristi za upravljanje infrastrukturom. Omogućava definisanje i upravljanje infrastrukturom putem koda (*IaC*). Kod predstavljaju konfiguracioni fajlovi napisani u *HashiCorp Configuration Language*-u (*HCL). Terraform* u velikoj meri olakšava i ubrzava upravljanje infrastrukturom za razliku od manuelnog upravljanja. Sve *terraform* datoteke imaju *.tf* ekstenziju i sadrže definicije resursa, njihove karakteristike, odnose sa drugim resursima kao i druge parametre koji opisuju stanje infrastrukture.

HashiCorp Configuration Language (*HCL*) predstavlja deklarativni jezik koji je posebno dizajniran za *Terraform.* Dizajniran je tako da bude veoma jednostavan za čitanje i pisanje i strukturiran je tako da podseća na *JSON* format za razmenu podataka.

4.1. SINTAKSA I OSNOVNE KOMANDE

Osnovni deo sintakse za definisanje i konfigurisanje resura su Terraform blokovi. Blokovi pružaju strukturiran način organizovanja i specificiranja komponenti. Neki od najvažnijih terraform blokova su *terraform, provider, resource, data, locals i module.*

Osnovne komande koje se u *Terraform*-u koriste za interakciju sa infrastrukturnim resursima su:

- *terraform init* – koristi se za inicijalizaciju radnog direktorijuma i povlačenje svih potrebnih *plugin*-ova i provajdera definisanih u konfiguraciji.
- *terraform plan* – kreira pregled inkrementalnih promena na osnovu promenjene konfiguracije. Analizira trenutno stanje infrastrukture i poredi ga sa željenim stanjem.
- *terraform apply* - koristi se za primenu promena opisanih u izvršnom planu *Terraform*-a. Vrši kreiranje, promenu ili brisanje postojećih resursa.
- *terraform destroy* - vrši uništavanje svih resursa definisanih u *Terraform* konfiguraciji. Ova komanda je nepovratna pa je treba oprezno koristiti.

4.2. *TERRAFORM STATE FILE*

Terraform state file predstavlja fajl u kojem se čuva trenutno stanje infrastrukture. Ovaj fajl omogućava praćenje metapodataka i olakšava radnje poput primene

novog koda, uništavanje postojeće infrastrukture ili postavljanja novih resursa. Može se čuvati lokalno, ali se preporučuje udaljeno skladištenje na takozvanom t*eraform state backend*-u. Na *AWS*-u kao *state backend* koristi se *AWS S3 bucket*.

5. BEZBEDNOST NA *AWS*-U

Prva linija odbrane kada je u pitanju bezbednost na *cloud*-u predstavlja filtriranje ulaznog i izlaznog saobraćaja *VPC*-a. *AWS* nudi nekoliko servisa za filtriranje ulaznog i izlaznog saobraćaja kao što su sigurnosne grupe, *Network Access Control List i VPC Flow Logs* za praćenje logova:

- sigurnosne grupe – virtuelni *firewall* koji kontroliše sav ulazni i izlazni saobraćaj. Saobraćaj se kontroliše tako što se definišu ulazna i izlazna pravila. Sigurnosna grupa se može povezati sa različitim resursima kao što su *EC2*, *RDS* ili *AWS Load Balancer*
- network access control list – dodatni sloj sigurnosti koji predstavlja virtuelni *firewall* za kontrolisanje ulaznog i izlaznog saobraćaja podmreža u *VPC*-u. Pravila koja se definišu veoma su slična onim kod sigurnosnih grupa osim što nemaju stanje (*stateless*) i zahtevaju posebno kreiranje ulaznog i izlaznog pravila kako bismo dozvolili određenu komunikaciju.
- *VPC* flow logs - omogućava praćenje toka saobraćaja unutar *VPC*-a. Može se kreirati za ceo VPC, podmrežu ili mrežni interfejs.

5.1. *WEB APPLICATION FIREWALL (WAF)*

Web Application Firewall (*WAF*) predstavlja servis koji ima ulogu da zaštiti vec aplikacije od uobičajenih napada i botova. Pomaže u kreiranju listi za kontrolu pristupa i raznovrsnih pravila u vidu uslova koja se primenjuju protiv sigurnosnih pretnji. Neki od najčešćih napada od kojih nudi zaštitu uključuju *SQL Injection, Cross-Site Scripting (XSS), DDoS,* razni skeneri i botovi. Osim ovih definisanih pravila, *AWS* nudi i setove upravljanih pravila koja možemo naći na *AWS Marketplace*-u. Jedan od najvažnijih i najpopularnijih setova pravila je *OWASP Top 10* koji predstavlja listu deset najčešćih sigurnosnih rizika na veb aplikacijama.

6. PROJEKTOVANJE I IMPLEMENTACIJA INFRASTRUKTURE

Projektovanje i implementaciju infrastrukture potrebno je započeti detaljnom analizom same aplikacije i njenih funkcionalnosti kako bi se na osnovu toga postavila odgovarajuća arhitektura. Osim toga, razmatranje budžeta, veličine tima koji radi na razvoju kao i analiza geografske lokacije krajnjih korisnika aplikacije uticaće na odabir *AWS* regiona kao i na inicijalan broj okruženja. U ovom slučaju to će biti us-east-1 region i tri okruženja: *development, staging i production.*

6.1. PRIPREMA ZA IMPLEMENTACIJU I NAJBOLJE PRAKSE

Implementacija je započeta konfigurisanjem *Terraform*-a. Za svako okruženje kreiran je S3 bucket koji će se koristiti kao *terraform remote backend* kao i *AWS DynamoDB* tabela koja se koristi za zaključavanje fajla u kojem se čuva trenutno stanje infrastrukture kako bi se izbegli konflikti. Za verzionisanje *Terraform* koda i efikasno upravljanje istim, koristiće se *Github*.

Prilikom projektovanja *VPC*-a, osnovne komponente infrastrukture, od ključnog je značaja pridržavati se najboljih praksi u samom startu kako bi se infrastruktura bez poteškoća skalirala u budućnosti. Takođe, određene konfiguracije na *VPC*-u nije moguće menjati nakon kreiranja. Neke od najboljih praksi predstavljaju:

- korišćenje najvećeg *CIDR* bloka i jedinstvenog opsega *IP* adresa
- korišćenje *Elastic IP* adresa
- korišćenje tagova
- korišćenje *VPC Peering*-a
- planiranje skalabilne infrastrukture u samom početku implementacije

6.2. ANALIZA KREIRANIH RESURSA

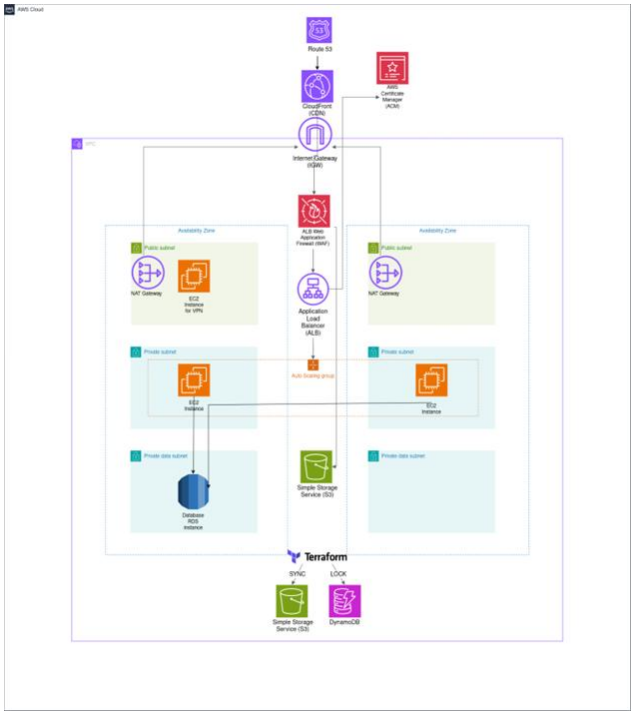
Za odabrani region najpre je kreiran *VPC*. Za produkciono okruženje odabran je 10.10.0.0/16 *CIDR* blok koji omogućava kreiranje 2^{16} odnosno 65536 hostova. *VPC* je podeljen na 6 podmreža, po jedna *public, private i private data*, u dve različite zone dostupnosti. Zatim su kreirane i ostale mrežne komponente kao što su tabele rutiranja, *NAT* i *Internet Gateway* za preslikavanje IP adresa i pristup internetu.

Sigurnosne grupe kreirane su uz poštovanje strogih pravila (*least privilege*, specifični opsezi *IP* adresa) koja nalažu da se svim resursima omogući samo neophodan saobraćaj. Dodatno je kreiran i *Wireguard VPN* server kako bi se *SSH* pristup serverima kao i pristup bazama podataka ograničio na privatnu mrežu. Osim sigurnosnih grupa, na infrastrukturi je radi bezbednosti implementiran i *WAF (Web Application Firewall*) koji štiti od najčešćih napada i sigurnosnih propusta.

Statički *front-end* veb sajt nalaziće se na *S3 bucket*-u koji je kreiran za njegovo hostovanje. Za distribuiranje statičkog sadržaja kreirana je *CloudFront* (*CDN*) distribucija.

Kako bi se obezbedila visoka dostupnost i horizontalna skalabilnost infrastrukture kreiran je *Application Load Balancer* (*ALB*). On kao *target* grupu ima *Auto Scaling* grupu koja, prateći gornje i donje parametre skaliranja, prilagođava broj *EC2* instanci na kojima je hostovan *back-end*.

Za relacione baze podataka koristi se *Amazon RDS* servis. Imajući u vidu da je projekat još uvek u ranoj fazi, nije korišćena *Multi AZ deployment* funkcionalnost kao ni *read* replike.



Slika 1. *Dijagram implementirane infrastrukture*

7. DISKUSIJA I ZAKLJUČAK

Primarni cilj rada bio je postavljanje visoko dostupne, skalabilne, fleksibilne kao i bezbedne infrastrukture koja će biti dobra osnova za dalji razvoj širokog spektra veb aplikacija.

Projekat je započet dizajniranjem *VPC* mreže vodeći računa o primeni svih najboljih bezbednosnih praksi. Nakon mreže, fokus je usmeren na dizajniranje i implementaciju resursa potrebnih za hostovanje veb aplikacija. U tu svrhu korišćeni su različiti servisi kao što su *S3, EC2, ALB i CloudFront* radi postizanja skalabilnosti, dostupnosti kao i optimizacije troškova.

Korišćenje *Terraform*-a, kao i samog pristupa u uspostavljanju celokupne infrastrukture putem koda je od presudne važnosti za efikasno upravljanje istom. Infrastrukturu je na ovaj način veoma lako replicirati na više različitih okruženja i izmene postaju veoma jednostavne uz minimiziranje mogućnosti za greške i narušavanje konzistentnosti okruženja.

Ova implementacija uspešno će zadovoljiti potrebe veb aplikacije, kako u početnoj fazi, tako i tokom skaliranja proizvoda. Naravno, prostora za napredak uvek ima. Jedna od potencijalnih oblasti unapređenja jeste korišćenje servisa kao što su *ECS (Elastic Container Service)* ili *EKS (Elastic Kubernetes Service)* za efikasnije upravljanje *docker* kontejnerima i ubrzanje *deployment*-a. Dodatno, uvođenje *CI/CD* pipeline-ova dodatno će poboljšati sistem obezbeđujući kontinuiranu isporuku i integraciju novog koda. Kako bi se greške u implementaciji i potencijalne pretnje otkrile na vreme, preporučuje se korišćenje *CloudWatch* servisa za praćenje logova uz kreiranje *CloudWatch* alarma.

8. LITERATURA

[1] AWS Global Infrastructure. https://aws.amazon.com/about-aws/global-infrastructure

[2] The NIST Definition of Cloud Computing. https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf

[3] Albert Anthony (2017) - Mastering AWS Security. Create and maintain a secure cloud ecosystem.

[4] Elastic Load Balancing (AWS). https://docs.aws.amazon.com/elasticloadbalancing/latest/application/introduction.html

[5] Neil David (2021) - AWS VPC Beginner to Pro - Virtual Private Cloud Tutorial. https://www.youtube.com/watch?v=g2JOHLHh4rI

[6] Amazon Relational Database Service (AWS). https://aws.amazon.com/rds

[7] Mohamed Jawad P (2018) medium.com - Amazon EC2: Auto Scaling. https://medium.com/@jawad846/amazon-ec2-auto-scaling-884ea50d2d

[8] What is Infrastructure as Code (IaC)? https://www.redhat.com/en/topics/automation/what-is-infrastructure-as-code-iac

[9] Web Application Firewall (AWS). https://aws.amazon.com/waf

[10] Terraform by HashiCorp. https://www.terraform.io

[11] Angelo Malatacca (2020) medium.com - AWS Terraform S3 and DynamoDB backend https://angelo-malatacca83.medium.com/aws-terraform-s3-and-dynamodb-backend-3b28431a76c1

Kratka biografija:

Denis Fruža rođen je u Zrenjaninu 1998. Diplomirao je na Fakultetu tehničkih nauka 2021. god. i tada upisuje master studije.

kontakt: denisfruza98@hotmail.com

Зборник радова Факултета техничких наука, Нови Сад

UDK: 4
DOI: https://doi.org/10.24867/31BE33Tepsic

ИМПЛЕМЕНТАЦИЈА ETHEREUM БАЗИРАНЕ МУЗИЧКЕ ПЛАТФОРМЕ

IMPLEMENTATION OF ETHEREUM-BASED MUSIC PLATFORM

Силвија Тепшић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *Задатак рада представља развој децентрализоване апликације за куповину, продају и репродукцију музике употребом Ethereum платформе. Циљ апликације је да демонстрира употребу blockchain базиране платформе која може да замени корпоративне посреднике*

Кључне речи: *blockchain, Ethereum, dApp, Solidity, музика*

Abstract – *The thesis deals with the development of a decentralized Ethereum-based application for purchasing, selling and playing music. The goal of the application is to demonstrate the use of a blockchain-based platform that can replace corporate intermediaries.*

Keywords: *blockchain, Ethereum, dApp, Solidity, music*

1. УВОД

У савременом свету корисници који желе да слушају музику и при том подрже извођаче углавном користе музичке платформе попут *iTunes*-а или *Amazon Music*-а. Да би се песма објавила на таквим платформама, она мора да прође кроз низ провера и мора да буде одобрена од стране таквих платформи. Издавачи такође морају и да верују платформама да су њихове песме складиштене на безбедном месту. Још једна мана таквих платформи је што узимају не тако мали проценат од продаје песама. Са циљем да се избегне потреба за коришћење посредничке платформе са комплексним процедурама и високим накнадама, развијена је музичка платформа *SongCloud.*

SongCloud представља децентрализовану апликацију која се ослања на *Ethereum* платформу и омогућава корисницима да издају, купују и репродукују музику. За потребе извршавања трансакција на платформи уводи и нову валуту - капљице.

2. BLOCKCHAIN

Да би се разумело функционисање *blockchain*-а, неопходно је да се разуме појам дистрибуираног система.

Дистрибуирани систем представља рачунарску парадигму где два или више чворова координисано сарађују како би постигли заједнички циљ. Моделован је тако да га крајњи корисници виде као јединствену логичку платформу [1].

Чвор представља индивидуалног члана (рачунар) у дистрибуираном систему. Чворови садрже меморију и процесор и могу међусобно да комуницирају путем порука. Чвор може да буде искрен, неисправан или злонамеран, као и да се понаша произвољно. У том случају се он зове византијски чвор. Назив потиче од приче о проблему византијских генерала [1].

Дизајн дистрибуираних система је поприлично изазован задатак јер је неопходно да се реше проблеми координације између чворова и отпорности на грешку, односно да чак и ако неки од чворова постану неисправни или се прекину линије комуникације, дистрибуирани систем би требао да настави успешно да функционише. Међутим, доказано је да дистрибуирани систем не може истовремено да поседује све три жељене особине: конзистентност, доступност и отпорност на партиције [1].

Blockchain представља дистрибуирану базу која постоји истовремено на више рачунара. Константно расте додавањем нових група забелешки односно блокова на њу. Сваки блок садржи временску ознаку и линк ка претходном блоку, тако да блокови заправо формирају ланац [2].

Кључна одлика *blockchain*-а је да је децентрализован, што значи да га не контролише један ентитет попут банке или владе. Уместо тога, одржава га мрежа рачунара који раде заједно да би валидирали и забележили трансакције. Овај приступ отежава модификацију података на *blockchain*-у од стране појединца, јер би то значило да тај појединац мора да убеди већину рачунара на мрежи да прихвате његове измене [3].

Постоји више различитих врста *blockchain*-а, укључујући јавни, приватни, конзорцијум и хибридни *blockchain*. Јавни *blockchain* је доступан свима и децентрализован, што значи да није контролисан од стране једног ентитета. Приватни *blockchain* је доступан само одређеној групи појединаца. Обично га користе организације које желе да им подаци буду приватни и безбедни. Конзорцијум *blockchain* је делимично децентрализован, пошто га контролише група организација, а не један ентитет. Обично се користи у индустријама где више странака треба да сарађују, попут финансијске индустрије. Хибридни *blockchain* комбинује аспекте јавног и приватног *blockchain*-а. Може да буде доступан јавно, али и да има неке елементе којима само одређени појединци или групе имају приступ [3].

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

Промене протокола *blockchain* мреже и структура података се називају рачвања (енгл. *forks*). Могу да се поделе у две категорије: *soft forks* и *hard forks* [5]. *Soft fork* представља промену у *blockchain* имплементацији која је компатибилна са претходним верзијама. *Hard fork* представља промену у *blockchain* имплементацији која није компатибилна са претходним верзијама.

3. ETHEREUM

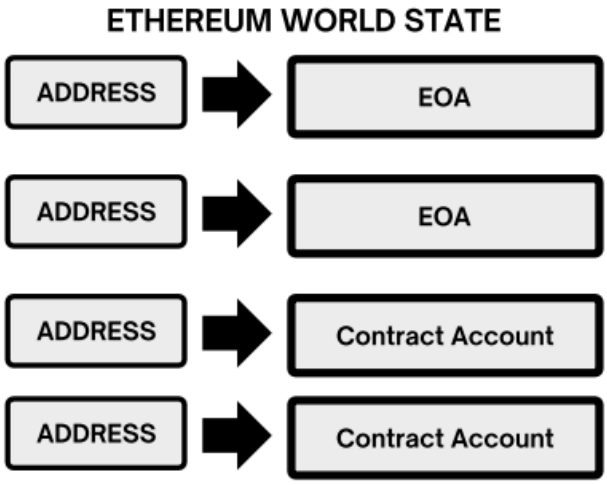
Ethereum представља децентрализовану *Peer-to-Peer* мрежу *Ethereum* клијената, који су покренути на чворовима. *Ethereum* клијент представља софтвер који може да верификује нове трансакције, извршава паметне уговоре и обрађује нове блокове ланца. Криптовалута која се користи на мрежи за плаћање трансакција и рачунарске обраде се назива *ether*, а званичан симбол јој је *ETH* [4].

3.1. Налози

Када корисник покрене *Ethereum* софтвер додељује му се идентитет у виду *Ethereum* налога са одређеним псеудонимом. Постоји и још једна врста налога, а то су налози уговора [6]. Налози у спољном власништву (енгл. *Externally Owned Accounts - EOA*) су кориснички налози који могу да креирају трансакције и којима се управља помоћу пара кључева, које *Ethereum* софтвер изгенерише. Налози уговора (енгл. *Contract accounts - CA*) представљају паметне уговоре који су испоручени на *Ethereum* и контролисани од стране њиховог интерног кода.

Стање *Ethereum*-а (енгл. *world state*) представља базу података која мапира *Ethereum* адресе на налоге у спољном власништву и налоге уговора. Уколико се било шта ново деси, стање *Ethereum*-а мора да се ажурира. Било да *EOA* пребацује средства другом *EOA* или интерагује са налогом уговора, долази до промене стања. Стање *Ethereum*-а се мења на сваких 12 секунди на основу свих трансакција које су корисници иницирали [6].

На слици 1 је дат приказ стања *Ethereum*-а.



Слика 1. *Ethereum стање [6]*

3.2. Паметни уговори и *Solidity*

Паметни уговори представљају непроменљиве рачунарске програме који се извршавају детерминистички у контексту *Ethereum* виртуелне машине као део *Ethereum* мрежног протокола, односно на децентрализованом *Ethereum* светском рачунару [7].

Децентрализована апликација (енгл. *Decentralised Application - dApp*) представља комбинацију паметног уговора са технологијама које нису део *blockchain*-а, како би се унапредило искуство корисника. Обично подразумева да постоји клијентски део у форми традиционалног веб сајта [6].

Solidity [8] је најпопуларнији програмски језик за писање паметних уговора. Он спада у статички типизиране језике, што значи да је потребно експлицитно дефинисати све типове података пре коришћења.

3.3. Трансакције

Постоје два главна типа трансакција у оквиру *Ethereum*-а. Први је позив поруке (енгл. *Message Call*). Позиви поруке креирани од стране *EOAs* подразумевају промену *Ethereum* стања. Други тип трансакције јесте креирање уговора (енгл. *Contract Creation*), који представља захтев да се дода нови налог уговора у *Ethereum* стање [6].

Да би се транксација реализовала у оквиру *Ethereum*-а, неопходно је да се плати одговарајућа накнада. За разумевање накнада за трансакције, неопходно је да се разуме разлика између цене гаса и накнаде за гас. у оквиру *Ethereum*-а циклуси обраде се изражавају помоћу мерне јединице која се назива гас. Гасни трошкови нису повезани са неком валутом, већ искључиво са бројем циклуса рачунарске обраде [6].

Свака јединица гаса која се потроши ће да резултује додатном накнадом за гас. Накнада за гас се изражава помоћу гвеја (енгл. *gwei*), а један гвеј вреди 0.0000000001 *ETH*. Накнада за гас се састоји из два дела: основне накнаде (енгл. *base fee*), која се алгоритамски рачуна на основу вредности гвеја, као и накнаде за приоритет (енгл. *priority fee*), која представља додатни износ који корисник може да плати како би његова трансакција била брже прихваћена [6].

3.4. Консензусни слој

Консензусни механизам који се користи у оквиру *Ethereum*-а се зове доказ улога (енгл. *Proof of Stake – PoS*). Заснива се на постојању валидатора чија су задужења валидирање трансакција и креирање блокова. Да би корисник постао валидатор, неопходно је да уложи 32 *ETH* на депозитни уговор као улог и да користи потпуни чвор у комбинацији са клијентом за валидацију. Постојање улога је битно како би се спречили Сибил напади [6].

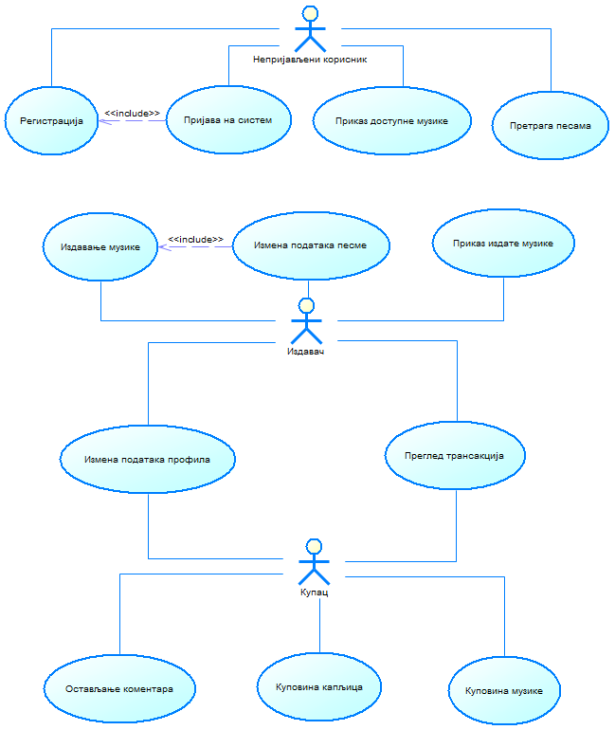
3.5. *IPFS*

IPFS (*Inter-Planetary File System*) представља децентрализовани складиштни систем са адресираним садржајем који дистрибуира сачуване објекте између учесника у *P2P* мрежи. Адресирани садржај подразумева да је сваки део садржаја (датотека) хеширан и да се та хеш вредност користи за идентификовање те датотеке. Могуће је добити било коју датотеку од било ког *IPFS* чвора захтевањем њеног хеша [7].

4. СПЕЦИФИКАЦИЈА

4.1. Спецификација захтева

На слици 2 је приказан дијаграм случајева коришћења. Он представља графичку репрезентацију корисника система (актора) и њихове могуће интеракције са системом. Као што се може видети на дијаграму, апликацију могу користити три типа корисника: непријављени корисник, издавач и купац.



Слика 2. *Дијаграм случајева коришћења*

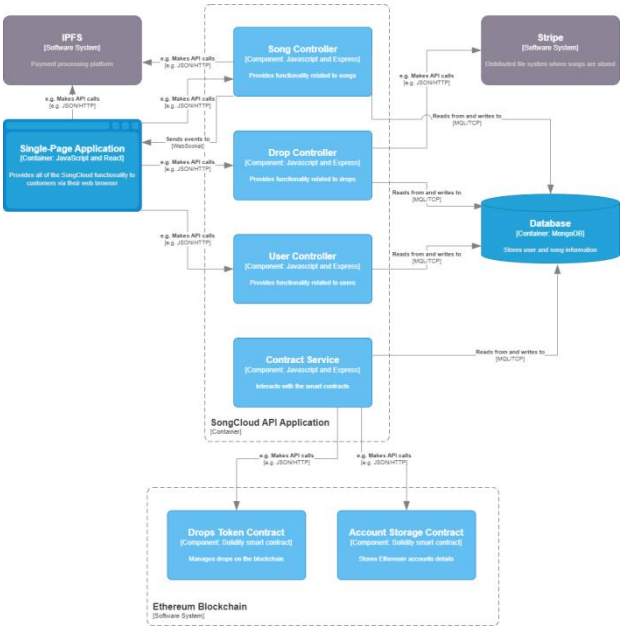
4.2. Спецификација система

Архитектура система *SongCloud* платформе се састоји од једностраничне апликације (енгл. *Single-page application*) са клијентске стране, *SongCloud API* апликације и паметних уговора са серверске стране, као и базе за трајно складиштење података.

Клијентска апликација омогућава корисницима интеракцију са системом и директно приступа датотекама песама на *IPFS*, што омогућава корисницима да их одслушају у претраживачу.

Главне делове *SongCloud API* апликације чине контролери, као и сервис који омогућава интеракцију са паметним уговорима – *Contract Service*. Основна улога контролера је да прихватају захтеве од клијентске апликације и да врате одговор који садржи информације о траженом ресурсу. *Drop Controller* врши и интеракцију са процесором плаћања *Stripe API*-ом, који омогућава реализацију транксација у еврима. Посебну улогу има *Song Controller* који може са серверске стране да иницира комуникацију са клијентом употребом *Web Socket* технологије. Поред тога *Song Controller* је задужен и да иницира слање датотека песама на *IPFS*. Ради чувања података о корисницима и песмама, *SongCloud API* апликација интерагује са базом података.

На слици 3 је приказана архитектура система *SongCloud* платформе.



Слика 3. *Дијаграм компоненти*

5. ИМПЛЕМЕНТАЦИЈА

5.1. Паметни уговори

У оквиру развоја система креирана су и коришћена два паметна уговора: *AccountStorage.sol* и *DropsToken.sol*.

AccountStorage.sol се користи превасходно за управљање корисничким идентитетом и власништвом. Он врши мапирање *Ethereum* адреса налога корисника на њихове *email* адресе, као и мапирање адреса налога корисника на песме које они поседује односно на хеш песме и њену цену.

DropsToken.sol се користи превасходно за управљање капљицама на платформи. *DropsToken.sol* уговор имплементира *ERC20* стандард, који представља протокол за имплементацију паметног уговора који ће се користити као токен на *Ethereum* мрежи. Он прописује одређене функционалности које паметни уговор мора да садржи. Овај уговор садржи и мапирање *Ethereum* адреса налога корисника на количину токена који тај налог поседује. На овај начин је онемогућена неовлашћена измена финансијског стања на рачуну корисника.

5.2. *SongCloud API* апликација

SongCloud API апликација представља главни серверски део платформе. Она комуницира са паметним уговорима испорученим на *Ethereum* мрежу, комуницира са базом података, као и са клијентским делом платформе.

Позиви ка паметним уговорима се реализују у *Contract Service*-у. У оквиру овог сервиса имплементиране су функционалности добављања и испоруке паметних уговора, креирања *Ethereum* налога корисника, функционалности за управљање капљицама, као и функционалности за управљање подацима о песмама на *blockchain*-у.

У оквиру *SongCloud API* апликације постоје три контролера: *User Controller*, *Drop Controller* и *Song*

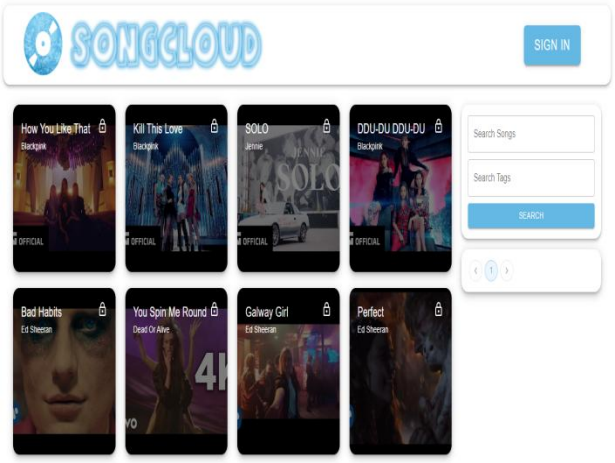
Controller. Преко њих се врши комуникација са клијентом, а уколико је за реализацију њихових функционалности неопходна комуникација са *blockchain* –ом, у њима се позива и *Contract Service*.

User Controller је задужен да реализује креирање налога корисника на платформи, пријаву корисника на платформу, добављање и измену корисничких података. У оквиру *Drop Controller*-а обавља се куповина капљица и добављање корисникових трансакција на платформи. У оквиру *Song Controller*-а извршавају се функционалности објаве, добављања, измене, куповине, коментарисања и претраге песама.

6. ДЕМОНСТРАЦИЈА

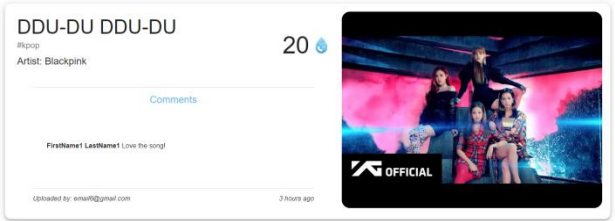
У овом поглављу описан је сценарио коришћења *SongCloud* апликације од стране непријављеног корисника.

Када корисник приступи *SongCloud* апликацији, прикаже му се почетна страница (слика 4) са свим доступним песмама на платформи. На њој се налази и навигациони бар, који омогућава кориснику да се креће кроз апликацију, као и форма за претрагу, помоћу које корисник може да претражује песме на основу назива и ознака песме.



Слика 4. Почетна страница

Притиском на жељену песму из листе песама, отвара се страница која садржи додатне информације о песми. Осим назива и извођача песме, видљиви су и подаци о томе кад је песма издата и од стране кога, цена песме у изражена у капљицама, ознаке песме, као и коментари о песми, уколико постоје. На слици 5 се може видети како изгледа детаљнији приказ песме.



Слика 5. *Приказ песме*

7. ЗАКЉУЧАК

Пројекат представља музичку платформу која се извршава на *Ethereum* мрежи. Тренутно имплементиране функционалности апликације су издавање, куповина и репродукција музике, остављање коментара о песмама у реалном времену, измена информација о издатим песмама, куповина капљица помоћу картице, преглед трансакција, креирање, приказ и измена профила корисника.

Пројекат је замишљен и као прототип, који треба да покаже могућности *blockchain* технологије, односно да је могуће променити тржиште музичке индустрије избацивањем корпоративних посредника који узимају велики проценат од продаје песама, а имају и потпуну контролу над датотекама и подацима корисника којима такође могу да тргују.

За имплементацију система коришћене су модерне технологије, што представља добру основу за даљи развој. Могућ је развој у правцу повећања децентрализованост апликације, отклањања ограничења на постојећим функционалностима или развојем нових.

8. ЛИТЕРАТУРА

[1] *I. Bashir, "Mastering Blockchain"*, 2023
[2] *A. Banafa, "Introduction to Blockchain Technology"*, 2023
[3] *V. Krishnan, "The Essential Guide to Web3"*, 2023
[4] *X. Wu, Z. Zou, D. Song, "Learn Ethereum"*, 2023
[5] *D. Yaga, P. Mell, N. Roby, K. Scarfone, "Blockchain Technology Overview"*, 2018
[6] *P. Dylan-Ennis, "Absolute Essentials of Ethereum"*, 2024
[7] *A. M. Antonopoulos, Dr. G. Wood, "Mastering Ethereum"*, 2019

Кратка биографија:

Силвија Тепшић рођена је у Руми 1999. године. Дипломира на Факултету техничких наука 2022. године, након чега исте године уписује мастер академске студије на студијском програму Рачунарство и аутоматика, смер Електронско пословање.

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.7
DOI: https://doi.org/10.24867/31BE34Mihajlovic

INTEGRACIJA V2X TEHNOLOGIJE U PARKING SISTEME, RAZVOJ I IMPLEMENTACIJA WEB-APLIKACIJE

INTEGRATION OF V2X TECHNOLOGY IN PARKING SYSTEMS, DEVELOPMENT AND IMPLEMENTATION OF A WEB APPLICATION

Stefana Mihajlović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ENERGETIKA, ELEKTRONIKA I TELEKOMUNIKACIJE

Kratak sadržaj – *Cilj ovog master rada je istražiti integraciju V2X tehnologije u parking sisteme kako bi se unapredila efikasnost i korisničko iskustvo. Rad će obuhvatiti analizu trenutnih izazova u upravljanju parking mestima i predložiti rešenja kroz razvoj web-aplikacije. Aplikacija će omogućiti korisnicima pristup informacijama o dostupnosti parking mesta u realnom vremenu. Takođe, fokusiraće se na tehničke aspekte implementacije, uključujući izbor tehnologija i alata.*

Ključne reči: *V2X tehnologija, web-aplikacija*

Abstract – *The aim of this master's thesis is to explore the integration of V2X technology into parking systems to enhance efficiency and user experience. The study will address current challenges in parking management and propose solutions through the development of a web application. This application will provide users with real-time information about the availability of parking spaces. Additionally, it will focus on the technical aspects of implementation, including the selection of technologies and tools.*

Keywords: *V2X technology, Web application*

1. UVOD

U današnjem dinamičnom svetu, urbani prostori se suočavaju sa brojnim izazovima, uključujući sve veću gustinu saobraćaja, nedostatak parking mesta i potrebu za efikasnijim upravljanjem resursima. Razvoj tehnologija kao što je V2X (Vehicle-to-Everything) predstavlja značajan korak napred u rešavanju ovih problema.

Cilj ovog master rada je istražiti integraciju V2X tehnologije u parking sisteme i razvoj web-aplikacije koja će korisnicima omogućiti lakše upravljanje parking mestima. Ova aplikacija će koristiti realne podatke o dostupnosti parking mesta i pružiti korisnicima informacije u realnom vremenu, čime će se smanjiti vreme potrage za parkingom i poboljšati ukupno korisničko iskustvo.

Kroz analizu postojećih rešenja i razvoj inovativnih funkcija, ovaj rad će doprineti razumevanju potencijala V2X tehnologije u modernizaciji parking sistema.

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

Na kraju, očekuje se da će rezultati ovog istraživanja pružiti značajne uvide u budućnost upravljanja parking mestima i doprineti razvoju pametnih gradova koji su prilagođeni potrebama svojih stanovnika.

2. V2X TEHNOLOGIJA

Bežična komunikacija postala je ključna tehnologija za razvoj vozila narednih generacija. Cilj aktivnosti 3GPP (3rd Generation Partnership Project) je unapređenje LTE sistema kako bi se omogućila komunikacija između vozila, pešaka i infrastrukture. Ova komunikacija omogućava razmenu poruka za pomoć i bezbednost na putevima, kontrolu toka saobraćaja i pružanje različitih obaveštenja o saobraćaju.

V2X je sistem komunikacije koji prenosi informacije iz vozila ka ostalim delovima saobraćaja. Koristeći spektralno efikasan air interfejs, isplativo postavljanje mreže i podršku raznovrsnim tipovima komunikacije, LTE sistemi sa odgovarajućim unapređenjima mogu postati ključni pokretač V2X usluga. V2X tehnologija pozitivno utiče na efikasnost saobraćaja, na primer, predlaže vozačima alternativne rute kako bi izbegli gužve u određenim delovima grada. Takođe, upozorava vozače na stanje parkinga i identifikuje slobodna parking mesta.

Krajnji cilj u razvoju automobilske industrije jeste maksimalna bezbednost svih učesnika u saobraćaju. U slučaju saobraćajnih nezgoda, omogućava se automatsko uspostavljanje poziva za pomoć. Pored toga, mogu se prenositi i druge važne informacije koje se prosleđuju drugim učesnicima u saobraćaju, kao i uređajima koji su deo saobraćajne infrastrukture.

Nove tehnologije omogućavaju povezivanje i komunikaciju u saobraćaju. Takođe, 4G arhitektura ima efikasne tehnike rutiranja i rešenja u pogledu frekvencijskih opsega. Uz pomoć 4G i 5G umrežavanja, zahvaljujući malom kašnjenju, pouzdanosti konekcije i velikoj brzini protoka podataka, moguće je implementirati C-V2X (Cellular Vehicle-to-Everything) [1,2].

2.1. LTE-V2X KOMUNIKACIONE PORUKE

LTE-V2X komunikacione poruke igraju ključnu ulogu u razmeni informacija između vozila i drugih učesnika u saobraćaju. Ove poruke sadrže važne podatke o vozilu, uključujući trenutnu lokaciju (koordinate), brzinu, ubrzanje, pravac i smer kretanja. Prema ETSI (European Telecommunications Standards Institute), u okviru LTE-V2X postoje dva osnovna tipa komunikacionih poruka:

- CAM (Cooperative Awareness Messages): Ove poruke pružaju informacije o prisutnosti vozila i

njegovim karakteristikama, omogućavajući drugim učesnicima u saobraćaju da budu svesni okoline. CAM poruke se šalju periodično i pomažu u stvaranju svesti o situaciji na putu.

- DENM (Decentralized Environmental Notification Messages): Ove poruke se koriste za obaveštavanje o specifičnim događajima ili promenama u okruženju, kao što su saobraćajne nesreće ili prepreke na putu. DENM poruke su obično aktivirane određenim događajem i omogućavaju brzu reakciju vozila i vozača.

Komunikacione poruke poput CAM i DENM su od suštinskog značaja za V2X tehnologiju jer omogućavaju brzu razmenu informacija koja može značajno poboljšati bezbednost na putevima. Na primer, kada vozilo primi CAM poruku od drugog vozila koje se približava, vozač može pravovremeno reagovati kako bi izbegao potencijalnu nesreću. Slično tome, DENM poruke mogu upozoriti vozače na opasnosti ili promene u saobraćaju, čime se smanjuje rizik od nezgoda [1, 2].

2.2. C-V2X TEHNOLOGIJA

C-V2X (Cellular Vehicle-to-Everything) predstavlja naprednu komunikacionu tehnologiju koja omogućava vozilima da razmenjuju informacije sa drugim vozilima, pešacima i infrastrukturom. Ova tehnologija je definisana standardom 3GPP i koristi PC5 interfejs u opsegu od 5,9 GHz za slanje i primanje poruka. Kao osnovna tehnologija, C-V2X se oslanja na LTE, pri čemu fizički sloj koristi SC-FDMA i strukturu frejma koja je slična onoj u LTE uplink komunikaciji.

C-V2X obuhvata dva ključna komunikaciona linka:

- Vehicle-to-Network (V2N): Ova veza se oslanja na postojeću mobilnu mrežu (Uu) za komunikaciju.
- Vehicle-to-Vehicle (V2V): Ova veza omogućava direktnu komunikaciju između vozila.

Važno je napomenuti da svi V2V slučajevi zahtevaju direktnu vezu između učesnika u saobraćaju. C-V2X se posebno fokusira na ovu direktnu komunikaciju putem PC5 interfejsa. Razlikovanje između uređaja koji upravljaju mrežnom komunikacijom i direktne veze je ključno za osiguranje sigurne kontrole vozila.

C-V2X funkcioniše u dva režima raspoređivanja prenosa:

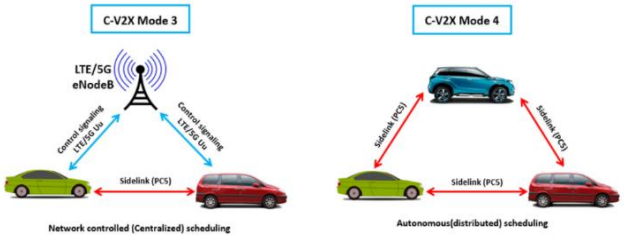
- Mode 3: Ovaj režim koristi mobilnu mrežu za alokaciju i upravljanje resursima.
- Mode 4: Ovaj režim omogućava rad van pokrivenosti ili bez SIM kartice, čime C-V2X postaje potpuno autonoman i nezavistan od mobilne mreže.

Sve primene C-V2X koriste samo Mode 4. Pre nego što Mode 3 postane održiv, potrebno je rešiti različite izazove, uključujući koštanje usluga i izvodljivost algoritama.

C-V2X igra ključnu ulogu u bezbednosti saobraćaja i sprečavanju nezgoda, posebno kada su u pitanju ranjivi učesnici poput pešaka, biciklista i motociklista. Integracija PC5 interfejsa u pametne telefone dodatno poboljšava funkcionalnost ove tehnologije [3, 4].

3. SLOJ APLIKACIJE

Sloj aplikacije predstavlja najviši nivo u OSI referentnom modelu i TCP/IP modelu, igrajući ključnu ulogu u funkcionisanju mrežnih aplikacija. Ovaj sloj opisuje interakciju aplikacija sa servisima i protokolima nižih slojeva, a protokoli u ovom sloju omogućavaju razmenu podataka između programa na predajnoj i prijemnoj strani



Slika 1. *Režimi raspoređivanja prenosa [5]*

.

Aplikacijski sloj je najbliži krajnjem korisniku, pružajući mrežne usluge aplikacijama. On uspostavlja komunikaciju, sinhronizuje procedure oporavka u slučaju grešaka i kontroliše integritet podataka. U ovom sloju postoje programi kao što su upravljanje bazama podataka, elektronska pošta i servisi za razmenu datoteka. Takođe, koristi DNS protokol za prevođenje imena domena u IP adrese.

Aplikacijski sloj je neophodan za komunikaciju između korisnika i računara. Omogućava nam da tražimo informacije ili ih delimo, kao i da koristimo e-mail za komunikaciju. Na internetu možemo pronaći praktično sve informacije koje nas interesuju, a multimedija nam pruža zabavu u svakom trenutku.

Sloj aplikacije je ključna komponenta u mrežnoj arhitekturi, jer povezuje korisnike sa različitim uslugama i resursima. Bez ovog sloja, komunikacija između ljudi i računara ne bi bila moguća na današnjem nivou. Razvoj novih protokola i usluga u ovom sloju kontinuirano poboljšava naše iskustvo na mreži, čineći ga važnim za budućnost digitalne komunikacije [6].

3.1. WEB-APLIKACIJA

Web programiranje obuhvata kreiranje web sajtova i aplikacija koristeći tehnologije kao što su HTML, CSS i JavaScript. Web sajtovi su obično zbir statičkih stranica koje pružaju informacije, dok su web aplikacije interaktivnije i omogućavaju korisnicima unos i obradu podataka, kao što su Gmail ili Google Docs.

Statični sajtovi prikazuju iste informacije svim korisnicima, dok dinamički sajtovi generišu različit sadržaj u zavisnosti od korisničkog unosa ili interakcije. Na primer, dinamički sajt može prilagoditi prikaz informacija na osnovu preferencija korisnika ili prethodnih interakcija.

Za početak učenja web programiranja, preporučuje se fokusiranje na HTML za strukturu, CSS za stilizaciju i JavaScript za dodavanje interaktivnosti. Nakon ovih osnovnih tehnologija, korisno je preći na backend tehnologije poput PHP-a ili Node.js-a kako bi se kreirali dinamički sajtovi.

Za razliku od desktop aplikacija koje se instaliraju na lokalnom računaru, web aplikacije se izvršavaju na udaljenim serverima, a korisnici im pristupaju putem interneta. Važno je napomenuti da svaki web sajt ne mora biti i web aplikacija; web sajt može biti jednostavna kolekcija informacija bez interaktivnosti.

Web sajt se sastoji od povezanih web stranica koje su dostupne globalno putem interneta i imaju jedinstveno ime

domena. Ove stranice mogu biti smeštene na jednom ili više web servera. S druge strane, web aplikacije su softver koji se takođe može globalno pristupiti putem interneta, ali pružaju dodatne funkcionalnosti koje omogućavaju korisnicima interakciju s podacima [7, 8].

3.2. ASP.NET CORE

Jedna od najpoznatijih platformi za postavljanje i odgovaranje na pitanja vezana za programiranje, Stack Overflow koristi ASP.NET Core kao deo svoje infrastrukture. Ova platforma koristi snagu ASP.NET Core-a kako bi obezbedila brze reakcije i efikasno upravljanje velikim količinama podataka i korisnika. Korišćenje ovog framework-a omogućava im da optimizuju performanse i poboljšaju korisničko iskustvo.
Microsoft Teams je alat za kolaboraciju i komunikaciju unutar kompanija koji koristi više tehnologija, ali ASP.NET Core igra ključnu ulogu u izgradnji određenih delova aplikacije. Ovaj framework omogućava skalabilnost i pouzdanost, što je od suštinskog značaja za softverska rešenja koja zahtevaju visoku dostupnost i efikasnost.
Još jedan primer web aplikacije koja koristi ASP.NET Core je Reddit, popularna platforma za razmenu informacija i diskusiju. Reddit koristi ASP.NET Core za svoje API-je, što im omogućava da efikasno rukovode velikim brojem zahteva od korisnika širom sveta. Ovaj framework pomaže u održavanju brzine i efikasnosti aplikacije, čak i pod velikim opterećenjem.
Primeri pokazuju snagu i fleksibilnost ASP.NET Core-a u različitim kontekstima i industrijama. Programeri koji rade s ovim framework-om mogu razvijati aplikacije koje su ne samo funkcionalne, već i visoko optimizovane za rad u stvarnom svetu.
ASP.NET Core podržava različite arhitekture poput MVC (Model-View-Controller) i Razor Pages, što dodatno olakšava razvoj složenih web aplikacija. Takođe, pruža mogućnosti za integraciju sa raznim bazama podataka, autentifikaciju korisnika i podršku za rad sa API-jima.
Kroz raznovrsne primene u stvarnom svetu, ASP.NET Core se pokazao kao moćan alat za razvoj modernih web aplikacija. Njegova sposobnost da podrži visoke performanse, skalabilnost i sigurnost čini ga idealnim izborom za programere koji žele da kreiraju inovativna rešenja [9].

4. IMPLEMENTACIJA APLIKACIJE

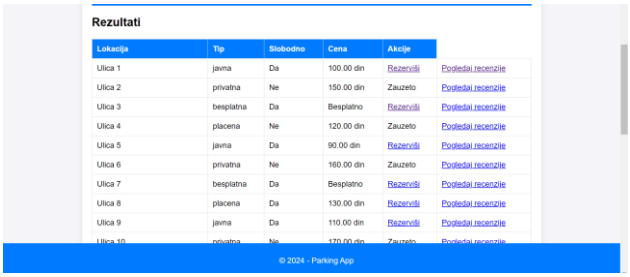
Problem: U urbanim sredinama često nedostaje informacija o slobodnim parking mestima. Vozači provode previše vremena u potrazi za dostupnim mestima, što dovodi do gužvi i povećava emisije štetnih gasova.
Rešenje: Razviti aplikaciju koja omogućava praćenje slobodnih parking mesta u realnom vremenu.

4.1. PROGRAMSKI JEZIK C#

C# je objektno orijentisan programski jezik koji je razvio Microsoft kao deo svoje .NET inicijative. Prvobitno ga je dizajnirao tim na čelu sa Andersom Hejlsbergom, a prvi put je predstavljen 2000. godine. Od tada, C# je prošao kroz brojne verzije, svaka sa novim unapređenjima i funkcionalnostima, čineći ga jednim od najpopularnijih jezika u industriji.
C# spada u grupu objektno orijentisanih jezika, što znači da se fokusira na upotrebu objekata i klasa za organizaciju koda. Ova paradigma omogućava programerima da kreiraju modularne i lako održive aplikacije. Sintaksa C# jezika je slična onoj u jezicima kao što su Java i C++, što olakšava prelazak između ovih jezika za programere koji su već upoznati sa njima.
C# je deo .NET ekosistema, što znači da koristi .NET framework ili .NET Core (sada poznat kao .NET 5+) za razvoj aplikacija. Ovi framework-ovi pružaju bogat skup biblioteka i alata koji olakšavaju razvoj i održavanje aplikacija. Visual Studio je najpopularnije razvojno okruženje (IDE) za rad sa C#, nudeći moćne alate za debagovanje, testiranje i upravljanje projektima.

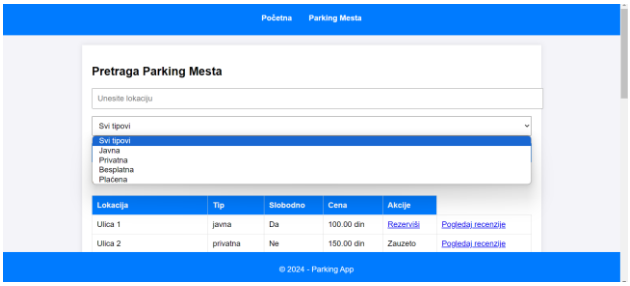
4.2. FUNKCIONALNOSTI APLIKACIJE

1. Prikaz slobodnih parking mesta u realnom vremenu
    - Mapiranje lokacija parking mesta
    - Vizuelna diferencijacija slobodnih i zauzetih mesta putem boja ili ikona



Slika 2. *Prikaz slobodnih parking mesta*

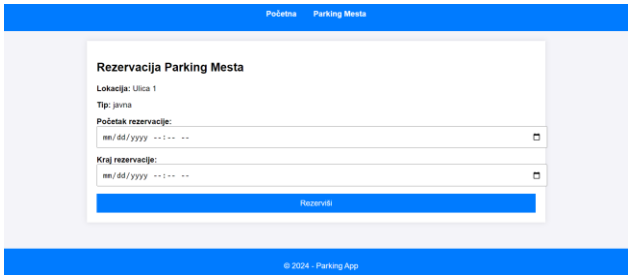
2. Pretraga i filtriranje parking mesta
    - Pretraga po lokaciji (naziv oblasti)
    - Filtriranje po vrsti parkinga (javna, privatna, besplatna, plaćena)



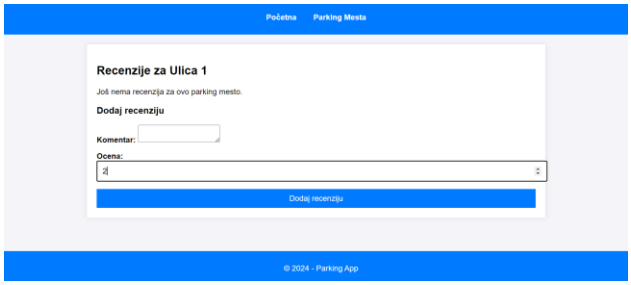
Slika 3. *Filtriranje parking mesta*

3. Rezervacija parking mesta
    - Mogućnost rezervacije slobodnog mesta na određeno vreme
    - Pregled i potvrda rezervacije
4. Korisničke recenzije i ocene
    - Opcija za ostavljanje recenzija i ocena za parking mesta

- Pregled recenzija i ocena drugih korisnika



Slika 4. *Rezervacija parking mesta*

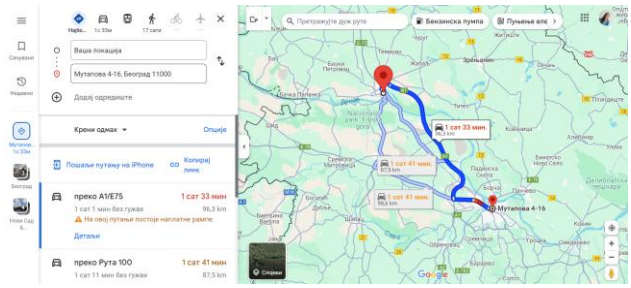


Slika 5. *Dodavanje korisničke recenzije*

5. Mape i navigacija
   - Integracija sa mapama za navigaciju do rezervisanog parking mesta

Slika 6. *Prikaz mape*

- Prikaz uputstava kako doći do parking mesta



Slika 7. *Navigacija do parking mesta*

5. ZAKLJUČAK

U savremenim urbanim sredinama, problem nedostatka informacija o slobodnim parking mestima može se značajno rešiti integracijom ASP.NET Core aplikacija sa V2X tehnologijom i različitim senzorima. Ova kombinacija ne samo da unapređuje efikasnost parkiranja, već i doprinosi smanjenju saobraćajnih gužvi i emisija štetnih gasova. ASP.NET Core omogućava razvoj moćnih RESTful Web API-ja koji mogu prikupljati i obrađivati podatke u realnom vremenu. Kroz korišćenje kontrolera, modela i CRUD operacija, programeri mogu izgraditi aplikacije koje efikasno upravljaju informacijama o slobodnim parking mestima. Na primer, aplikacija može prikazivati trenutnu dostupnost parkinga, omogućiti rezervaciju mesta i slati obaveštenja korisnicima. V2X tehnologija omogućava vozilima da komuniciraju sa infrastrukturom i drugim vozilima, čime se stvara mreža koja deli informacije o slobodnim parking mestima. Ova tehnologija može koristiti različite senzore, kao što su kamere sa prepoznavanjem oblika, GPS i ultrazvučni senzori. V2X tehnologija može poboljšati bezbednost na putevima tako što će vozila deliti informacije o potencijalnim opasnostima ili preprekama. Integracijom ASP.NET Core aplikacija sa V2X komunikacijom i senzorima, možemo stvoriti pametne urbane sisteme koji ne samo da olakšavaju parkiranje, već i doprinose održivijem razvoju gradova. Ovakva rešenja predstavljaju značajan korak ka modernizaciji urbanog transportnog sistema i unapređenju kvaliteta života u gradovima.

6. LITERATURA

[1] Hanbyul Seo, Ki-Dong Lee, Shinpei Yasukawa, Ying Peng, and Philippe Sartori, *LTE Evolution for Vehicle-to-Everything Services*, IEEE Communications Magazine (2016)

[2] How do the seven types of V2X connectivity work? (microcontrollertips.com), (pristupljeno u septembru 2024.)

[3] Yuanyuan Fan, Liu Liu, Shuoshuo Dong, Lingfan Zhuang, Jiahui Qiu, Chao Cai and Meng Song, *Network Performance Test and Analysis of LTE-V2X in Industrial Park Scenario*, Zhipeng Cai (2020)

[4] Alessandro Bazzi, Antoine O. Berthet, Claudia Campolo , Barbara Mavi Masini, Antonella Molinaro and Alberto Zanella, *On the Design of Sidelink for Cellular V2X*, A Literature Review and Outlook for Future, IEEE Access (2021)

[5] Amir Haider Malik, Seung-Hoon Hwang, *Adaptive Transmit Power Control Algorithm for Sensing-Based Semi-Persistent Scheduling in C-V2X Mode 4 Communication*, Electronics (2019)

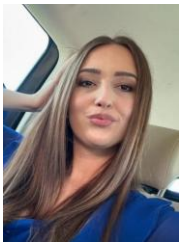
[6] Application Layer in OSI Model - GeeksforGeeks (pristupljeno u septembru 2024.)

[7] Introduction to Web Development with HTML, CSS, JavaScript | Coursera, (pristupljeno u septembru 2024.)

[8] Learn Web Development Basics – HTML, CSS, and JavaScript Explained for Beginners (freecodecamp.org), (pristupljeno u septembru 2024.)

[9] Overview of ASP.NET Core | Microsoft Learn, (pristupljeno u septembru 2024.)

Kratka biografija:



Stefana Mihajlović rođena je u Zvorniku 2000. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Informaciono-komunikacione tehnologije odbranila je 2024.god. kontakt: stefana.mihajlovic2@gmail.com

Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.7
DOI: https://doi.org/10.24867/31BE35Kovacev

ANALIZA PERFORMANSI LTE C-V2X TEHNOLOGIJE BAZIRANA NA KOMERCIJALNOJ EVALUACIONOJ PLATFORMI

PERFORMANCE ANALYSIS OF LTE C-V2X TECHNOLOGY BASED ON A COMMERCIAL EVALUATION PLATFORM

Emilija Kovačev, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ENERGETIKA, ELEKTRONIKA I TELEKOMUNIKACIJE

Kratak sadržaj – *U ovom radu prikazana je evolucija mobilnih telekomunikacionih sistema, sa posebnim osvrtom na LTE tehnologiju. Analiza LTE tehnologije pokazala je njenu ključnu ulogu u današnjim komunikacionim sistemima i u komunikaciji vozila sa svime. Posebno je opisan C-V2X i njegov značaj za saobraćaj. LTE-V2X, posebno Režim 4, omogućava direktnu komunikaciju između vozila bez potrebe za podrškom infrastrukture. Kroz test za C-V2X, analizirane su performanse i mogućnosti ove tehnologije, uz korišćenje Autotalks CRATON2 EVK platforme.*

Ključne reči: *4G mreža, C-V2X tehnologija, LTE V2X, pametna vozila, autonomna vožnja*

Abstract – *This paper presents the evolution of mobile telecommunication systems, with special reference to LTE technology. Analysis of LTE technology has shown its key role in today's communication systems and in vehicle communication with everything. C-V2X and its importance for traffic are especially described. LTE-V2X, especially Mode 4, enables direct communication between vehicles without the need for infrastructure support. Through a test for C-V2X, the performance and capabilities of this technology were analyzed, using the Autotalks CRATON2 EVK platform.*

Keywords: *4G network, C-V2X technology, LTE V2X, smart vehicles, autonomous driving*

1. UVOD

V2X (Vehicle to Everything) tehnologija predstavlja vrhunac razvoja modernih transportnih sistema, omogućavajući vozilima da komuniciraju sa svim elementima okruženja. Značaj V2X tehnologije leži u njenoj sposobnosti da stvori inteligentne transportne sisteme koji mogu značajno povećati sigurnost na putevima i efikasnost saobraćaja. Sa V2X tehnologijom, vozila mogu da primaju upozorenja o potencijalnim opasnostima na putu, čime se omogućava pravovremena reakcija vozača ili autonomnih sistema vozila. Implementacija V2X tehnologije predstavlja korak ka potpuno autonomnim vozilima.

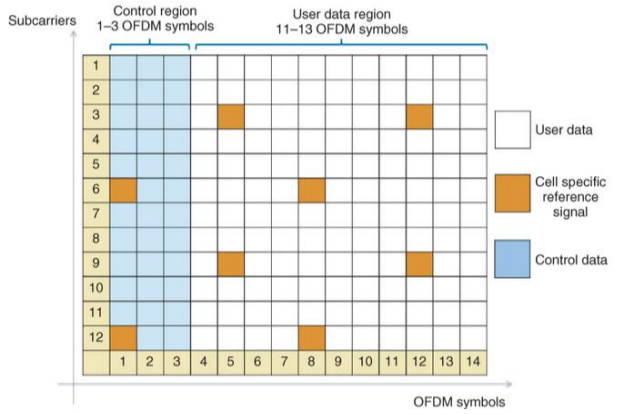
NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dejan Vukobratović, red. prof.

Ova tehnologija omogućava razvoj novih usluga i aplikacija koje mogu dodatno unaprediti korisničko iskustvo i podržati razvoj pametnih gradova.

2. LTE TEHNOLOGIJA

Pomoću LTE-a svet se konvergirao u jedinstvenu globalnu tehnologiju za mobilnu komunikaciju, koju koriste u suštini svi operateri mobilnih mreža i koja je primenljiva i na uparene i na neuparene spektre.
Od najvećeg značaja je njen PHY (fizički) sloj.
Vazdušni interfejs u LTE je za downlink zasnovan na OFDM tehnologiji višestrukog pristupa, dok je za uplink zasnovan na tehnologiji koja se naziva multipleksiranje sa frekvencijskom podelom jednog nosioca (Single Carrier Frequency Division Multiplexing – SC-FDM).
OFDM pruža značajne prednosti u poređenju sa ostalim tehnologijama i predstavlja veliki napredak u odnosu na prošlost. Neke od prednosti su visoka spektralna efikasnost, prilagodljivost za širokopojasni prenos, otpornost na međusimbolske smetnje uzrokovane višestrukim fedingom, podrška za MIMO (Multiple Input Multiple Output) i za frekventncijski domen.
Vremensko-frekvencijski prikaz OFDM-a pruža visoku fleksibilnost u alokaciji spektra i vremenskih frejmova za prenos. Fleksibilnost spektra omogućava različite frekvencijske opsege.
Takođe, LTE formira mali frejm kratkog trajanja (10 ms), kako bi učinio da kašnjenje bude najmanje moguće i time se obezbeđuje poboljšanja procena kanala u uređaju, pružajući baznoj stanici povratne informacije neophodne za prilagođavanje veze.
Standardi u LTE tehnologiji definišu raspoložive radio spektre za različite frekvencijske opsege. Jedan od zadataka standarda je savršeno usklađivanje sa ranijim mobilnim sistemima.
Osim unicast usluga, LTE standardi omogućuju i multicast usluge, tj. multimedijalne prenose sa velikom brzinom prenosa.
Prenos u LTE omogućava vremensku rezoluciju od 12 ili 14 OFDM simbola (u zavisnosti od cikličnog prefiksa), za svaki podfrejm od 1 ms.
Frekvencijska rezolucija, LTE omogućava niz blokova resursa u opsegu od 6 do 100 (u zavisnosti od propusnog opsega), od kojih svaki sadrži 12 podnosioca razmaknutih sa 15 kHz.
U mreži fizičkih resursa su sadžane tri vrste informacija.
U svakom elementu resursa se nalazi modulisani simbol korisničkih podataka ili referentnog ili sinhronizacionog

signala ili kontrolne informacije, koje potiču iz kanala višeg sloja. Na slici 1 su prikazane njihove lokacije (u mreži resursa), za jednostruki način rada.



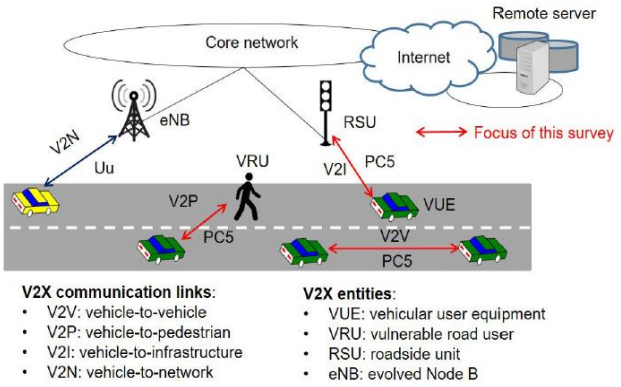
Slika 1 *Prikaz mreže resursa [1]*

Za jednostruki režim rada, informacije korisnika se prenose u korisničkim podacima i isporučuju se sa MAC sloja (Media Access Control) u PHY, u vidu transportnog bloka. Bazna stanica i mobilni uređaji generišu različite vrste referentnog i sinhronizacionog signala. iz razloga procene, merenja i sinhronizacije kanala.
Na kraju imamo različite vrste kontrolnih informacija, koje se dobijaju preko kontrolnih kanala i nose informacije koje su prijemniku potrebne da bi ispravno dekodovao signal.
LTE je bitno smanjio upotrebu namenskih kanala, u poređenju sa drugim 3GPP standardima i takođe se više se oslanja na zajedničke/deljene kanale.
Osobina LTE standarda je kreiranje efikasnijeg i modernizovanijeg steka i arhitekture protokola, u odnosu na prethodnike.
Šeme modulacije koje se koriste u LTE standardu uključuju QPSK (Quadrature Phase Shift Keying), 16QAM (16-Quadrature Amplitude Modulation) i 64QAM (64-Quadrature Amplitude Modulation) [1].

3. KOMUNIKACIJA VOZILA SA SVIME

V2X tehnologija formalno je poznata kao PC5 Sidelink, ali obično se naziva LTE-V2X ili Cellular Vehicle-to-Everithing (C-V2X). Sidelink preko PC5 interfejsa efektivno pretvara vozilo u RAN čvor proširujući vezu sa obližnjeg eNB-a na druge mobilne uređaje, je namenjena kao dodatak konvencionalnoj komunikaciji uplink/downlink veze preko Uu interfejsA [1]
U kontekstu V2X, sidelink se odnosi na direktnu međusobnu komunikaciju između vozila, između vozila i učesnika u saobraćaju (Vulnerable Road User – VRU), i između vozila i jedinica pored puta (Roadside Unit – RSU).
Preciznije, 3GPP koristi korisničku opremu vozila (Vehicular User Equipment – VUE) za označavanje vozila, i vozilo-vozilo (Vehicle-to-Vehicle – V2V), vozilo-pešak (Vehicle-to-Pedestrian – V2P), vozilo-infrastrukturu (Vehicle-to-Infrastructure – V2I), za adresiranje različitih vrsta veza , kao što je prikazano na slici 1.
Postoje prednosti LTE-V2X koje uključuju povećanje efikasnosti saobraćajnog toka, smanjenje uticaja na životnu sredinu i pružanje dodatnih usluga informisanja putnika. Kako bi se omogućila V2X tehnologija, postoje standardi za komunikaciju, koji su se razivijali prethodnih godina i koji koriste nelicencirani opseg od 5,9 GHz. Standardi su:

- Dedicated Short Range Communications – DSRC. Koristi standarde zasnovane na IEEE 802.11p i IEEE 1609 WAVE (Wireless Access in Vehicular Environments) za bezbednosne protokole i mrežni sloj.
- LTE-V2X. Može se smatrati alternativnom verzijom DSRC-a. Koristi PC5 interfejs sidelinka i IEEE 1609 WAVE standarde za bezbednosne protokole i mrežni sloj [2].



Slika 2 *V2X komunikacioni likovi, entiteti i radio interfejsi [2]*

3.1. Vrste V2X

Vehicle-to-Vehicle (V2V), koji se odnosi na direktnu razmenu podataka između vozila, zahvaljujući WiFi baziranim ili ćelijskim tehnologijama, zahvaljujući komunikaciji sidelink-a.

Vehicle-to-Infrastructure (V2I), kada vozilo prenosi i prima poruke ka/od infrastrukturnog čvora, koji se naziva jedinica na putu (RSU). RSU je fiksni uređaj koji se nalazi duž puta i služi kao fiksna pristupna tačka

Vehicle-to-Netvork (V2N), kada vozilo komunicira sa udaljenim serverom aplikacija koji hostuje uslugu zasnovanu na mreži, na primer putem mobilne veze.

Vehicle-to-Pedestrian (V2P), kada aplikacija zahteva razmenu podataka između vozila i ranjivih učesnika u saobraćaju, tj. pešaka i biciklista. V2P aplikacije obično imaju za cilj poboljšanje bezbednosti na putevima tako što će vozila i pešake učiniti svesnijim jedni o drugima, i pružanjem usluga kao što je izbegavanje sudara [1].

4. LTE-V2X

LTE-V2X je prvi put definisan u 3GPP izdanju 14 i predviđa dva različita interfejsa:

- LTE-PC5 režim: Režim podržava direktnu komunikaciju između čvorova preko PC5 interfejsa koristeći kanal sidelinka. Prisustvo entiteta eNodeB je opciono. PC5 interfejs se koristi za V2V komunikaciju u pristupu jedan-prema-više. Ovaj interfejs omogućava različitoj korisničkoj opremi (UE), tj. vozila, da komunicira direktno na 5,9 GHz DSCR opsegu preko kanala sidelinka.

- LTE-Uu režim. Primenljiv je za kratke udaljenosti. Informacije se prvo prenose do entiteta eNodeB preko kanala uzlazne veze putem izvornog čvora. Nakon toga, informacija se prenosi preko downlink-a do odredišnog čvora. Omogućava komunikaciju prema infrastrukturi, preko standardnog licenciranog spektra. [2]

Komunikacija preko PC5 može se desiti u sledeća dva režima:

- Režim 3 (kontrolisani). Planiranjem resursa i upravljanjem smetnjama preko PC5 interfejsa upravlja razvijeni NodeB (eNodeB) na centralizovan način, putem kontrolne signalizacije preko Uu interfejsa. Primjenjuje se samo pod pokrivenošću mreže.
- Režim 4 (autonomni). Ovakav način rada je ključan za suočavanje sa scenarijima van pokrivenosti. Vozila samostalno biraju i konfigurišu resurse i podkanale. Režim 4 ne zahteva nikakvu SIM karticu niti pretplatu [1].

5. PHY SLOJ LTE-V2X

PHY sloj LTE-V2X se zasniva na sledećim glavnim karakteristikama:

- višestruki pristup sa podelom frekvencije sa jednim nosiocem (SC-FDMA),
- turbo kodovi i
- HARQ retransmisija sa inkrementalnom redundantnošću.

Ove tri kombinovane karakteristike doprinose poboljšanju budžeta veze, što se direktno prevodi u proširenu pokrivenost pri fiksnoj snazi prenosa ili poboljšanu pouzdanost pri fiksnom dometu.

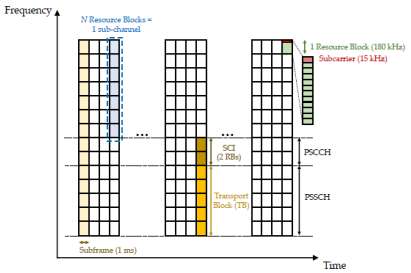
Na PHY sloju, granularnost dodele LTE-V2X resursa u vremenskom domenu data je vremenskim intervalom prenosa, koji odgovara jednom podfrejmu od 1 ms [2].

LTE-V2X koristi višestruki pristup sa frekvencijskom podelom jednog nosioca (SC-FDMA).

Podržana su dva propusna opsega kanala, 10 MHz i 20 MHz, i, u domenu frekvencije, svaki kanal je podeljen na blokove resursa (RB) širine 180 kHz. Svaki RB se sastoji od dvanaest podnosioca od 15 kHz.

U vremenskom domenu, kanal je podeljen na podfrejme od 1 ms, od kojih se svaki sastoji od dva slota od 0,5 ms i sadrži 14 OFDM simbola, od kojih se četiri koriste za prenos referentnih signala, što može pomoći u suzbijanju Doplerovog efekta. Devet simbola se koristi za prenos podataka, a poslednji simbol se koristi za prebacivanje između primljenog i prenetog preko podfrejma.

Ova vremensko-frekventna podela kanala je prikazana na slici 3 [3].



Slika 3 *Vremensko-frekventna podela LTE-V2X kanala, sa susednim zajedničkim fizičkim sidelinkom i fizičkim kontrolnim kanalom sidelinka [3]*

RB-ovi su sami grupisani u podkanale, sastavljene od više RB-a unutar istog podfrejma (stvarni broj je unapred konfigurisan i promenljiv) i koriste se za prenos podataka i kontrolnih informacija.

Prenos podataka je organizovan u transportnim blokovima (Transport Block – TB), predstavljenim grupama blokova resursa koji mogu da sadrže ceo paket. Veličina frejma, zajedno sa brojem RB-ova po podkanalu i izabranom šemom modulacije i kodovanja (Modulation and Coding Scheme – MCS), određuju veličinu TB-a, koji može da se prostire na jednom ili više podkanala. Na primer, nizak MCS će prouzrokovati da paket zahteva više RB-a nego kada je izabran visoki MCS. Alokacija resursa se dešava, preko podfrejmova (od 1 ms) i podkanala. Svaki TB je povezan sa takozvanim kontrolnom informacijom bočne veze (Scheduling Control Information – SCI), koja se prostire na dva RB-a i sadrži kontrolne informacije kao što su MCS, prioritet i izabrani interval rezervacije resursa (Retransmission Repetition Indicator – RRI).

Kako SCI sadrži ključne informacije za dekodovanje odgovarajuće TB, ona se uvek prenosi u istom podfrejmu i može se prenositi u RB-ovima koji su susedni onima rezervisanim za TB, kao na slici 3, ili u nesusednim [3].

LTE-V2X komunikacije sidelinka su uglavnom zasnovane na dva kanala koja se emituju u istom TTI od 1 ms, na različitim frekvencijama:

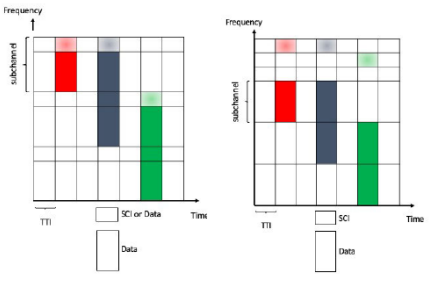
- fizički kontrolni kanal sidelinka (Physical Sidelink Control Channel – PSCCH), koji nosi kontrolne informacije (npr. prioritet, modulacija i šema kodovanja (Modulation and Coding Scheme – MCS), lokaciju frekvencijskog resursa, indeks retransmisije) u poruci informacija o kontroli sidelinka (SCI) i
- zajednički kanal fizičkog sidelinka (Physical Sidelink Shared Channel – PSSCH), koji prenosi podatke u transportnim blokovima (Transport Block – TB) [2].

Ovi kanali mogu biti susedni ili nesusedni.

Što se tiče MAC sloja, definisana su tri glavna pod-sloja: Protokol konvergencije paketnih podataka (Packet Data Convergence Protocol – PDCP), RLC (Kontrola radio veze) i MAC [3].

Svaki paket podataka se dodeljuje u jednom TTI i jednom ili više podkanala. Pridruženi SCI uvek zauzima 360 kHz, sa dve moguće konfiguracije, kao što je prikazano na slici 4. U susednoj konfiguraciji, SCI se prenosi unutar podkanala, tačnije na početku prvog dodeljenog podkanala.

U nesusednoj konfiguraciji, namenski resursi su rezervisani za SCI, izvan podkanala [2].

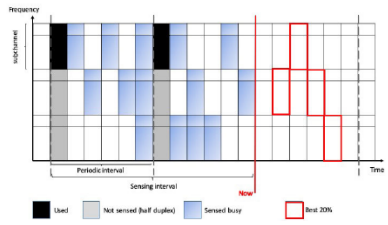


(a) Adjacent. (b) Non-adjacent.

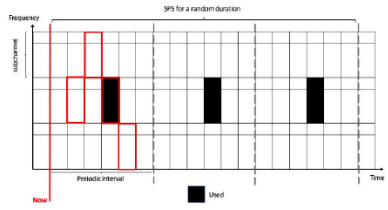
Slika 4 *Susedna (a) i nesusedna (b) alokacija kontrolnih informacija (SCI) u odnosu na vremensko-frekvencijsku mrežu resursa [2]*

6. C-V2X REŽIM 4

Režim 4 je uglavnom dizajniran za periodični saobraćaj i zasnovan je na polutrajnom rasporedu zasnovanom na sensingu. LTE-V2X režim 4 zahteva da vozilo samostalno dodeljuje i upravlja potrebnim resursima (tj. odgovarajućim podkanalima u vremenu), usvojena je šema SemiPersistent Scheduling (SPS) zasnovana na sensingu. Svakom vozilu se periodično dodeljuje ista grupa podkanala za dati interval. Skup podkanala potrebnih za prilagođavanje prenosa podataka označava se kao kandidatski resurs sa jednim podfrejmom (Cell Specific Subframe Resource – CSSR), a period se naziva interval rezervisanja resursa (Relative Resource Indicator – RRI).
Izbor CSSR-a, prikazanog na slici 5, vrši se na osnovu statusa resursa koji su praćeni tokom poslednjih 1s. Prema izdanju 14 3GPP standarda, svako vozilo treba da vrši sensing kanala u periodu od 1000 ms (koji se naziva Sensing Window i ekvivalentno je 1000 podfrejmova), kako bi se identifikovali potencijalni resursi jednog podfrejma koji ne bi trebalo da budu izabrani za prenos, jer se detektuju kao već zauzeti. Tokom intervala sensinga, čvor upoređuje primljenu snagu sa datim pragom sensinga.
Na osnovu informacija (vezanih za prošlost), bira 20% CSSR-a koji su manje verovatno zauzeti u (budućem) prozoru za izbor, tj. imaju niže nivoe snage primljenog referentnog signala [2].



(a) Sensing procedure.



(b) Resource selection and semi-persistent scheduling.

Slika 5 *Režim 4 selekcija resursa i procedura alokacije resursa [2]*

Među slobodnim resursima, vozilo nasumično bira između onih sa najnižim prosekom RSSI (Received Signal Strength Indicator). Selekcija se zatim održava za određeni broj prenosa, određen takozvanim Reselection Counter, koji obično pretpostavlja vrednosti od 5 do 15, iako zavisi od vrednosti RRI.
Očekuje se da će se prenos TB i SCI za brojače ponovnog izbora odvijati u periodičnosti koju definiše RRI. RRI, koji se prenosi u SCI, koristi se da informiše druga vozila o nameri da koriste iste izabrane resurse za sledeći prenos, koji će se desiti u vremenu RRI. Ovo se radi kako bi se izbeglo da drugi čvorovi izaberu iste resurse i izazovu kolizije paketa. RRI može da preuzme vrednost od 20 ms, 50 ms, 100 ms ili bilo koji umnožak od 100 ms do 1 sekunde. Posebna vrednost od 0 ms je rezervisana za slučaj u kojem vozilo neće rezervisati iste resurse za sledeći prenos TB i SCI.
Kako RRI može uticati na ukupne performanse C-V2X, on mora biti prilagođen što je više moguće karakteristikama željene komunikacije.
Ponovni izbor se takođe dešava u dva dodatna slučaja: ako je nova poruka prevelika za resurse koji su već rezervisani za odgovarajuću TB ili ako nova poruka ima kraći rok kašnjenja od trenutnog RRI, i mora se preneti pre vremena RRI [3].

7. KOMENTARI URAĐENOG TESTA

Prenos pokazuje visok stepen efikasnosti i stabilnosti, sa minimalnim brojem grešaka i resetovanja sistema. Iako su resursi ograničeni, sistem se nosi s tim situacijama bez većih problema. Većina poruka je uspešno obrađena, sa 100% uspešno poslatih poruka u nekoliko ključnih delova sistema, bez neuspešnih ili zakasnelih prenosa.
Merenja i segmenti alokacije su uglavnom stabilni, sa izuzetkom nekoliko promašenih merenja i manjih problema u alokaciji resursa. Vreme obrade komandi i resursa je u granicama očekivanog, bez značajnih odstupanja, što ukazuje na dobar odziv sistema.
Sve u svemu, performanse ovog prenosa su solidne, bez većih poteškoća u ključnim aspektima prenosa-

8. ZAKLJUČAK

LTE je tehnologija četvrte generacije koja omogućava velike brzine prenosa podataka i malo kašnjenje.
C-V2X omogućava komunikaciju vozila sa drugim vozilima, infrastrukturom i pešacima radi poboljšanja bezbednosti i podrške autonomnoj vožnji.
Test koji je rađen u ovom radu pokazuje da testirani uređaj pokazuje dobre performanse u laboratorijskim uslovima.

9. LITERATURA

[1] Houman Zarrinkoub, Understanding LTE with MATLAB, Wiley (2014)
[2] Alessandro Bazzi, Antoine O. Berthet, Claudia Campolo , Barbara Mavi Masini, Antonella Molinaro and Alberto Zanella, On the Design of Sidelink for Cellular V2X: A Literature Review and Outlook for Future, IEEE Access (2021)
[3] Francesco Raviglione, Open Platforms for Connected Vehicles, Politecnico di Torino (2022)

Kratka biografija:

Emilija Kovačev rođena je u Novom Sadu 1999. godine. Osnovne akademske studije je završila na Fakultetu tehničkih nauka iz oblasti Elektrotehnika i računarstvo – Telekomunikacioni sistemi. Master studije upisala je 2023. godine na smeru Informaciono-komunikacione tehnologije.
kontakt:
ekovacev99@gmail.com