Interoperabilnost interneta stvari

Rene, Zamostni

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Rene Zamostni

INTEROPERABILNOST INTERNETA STVARI

DIPLOMSKI RAD

Varaždin, 2020.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Rene Zamostni

Matični broj: 43358/14–R

Studij: Organizacija poslovnih sustava

INTEROPERABILNOST INTERNETA STVARI

DIPLOMSKI RAD

Mentor:

Prof. dr. sc. Nenad Vrček

Varaždin, srpanj 2020.

Rene Zamostni

Izjava o izvornosti

Izjavljujem da je moj završni/diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor/Autorica potvrdio/potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

Rad proučava teorijsku osnovu Interneta stvari, kroz prikaz njegovog razvoja, prisutnih tehnologija, izazova, mogućih primjena i vizija za budućnost. Praktični dio rada čini model prometnog sustava koji čini više uređaja koji međusobno komuniciraju, tvoreći tako cjelinu po uzoru na IoT. Posebni naglasak unutar praktičnog dijela stavljen je na interoperabilnost kao osnovnu osobinu uređaja unutar IoT sustava.

Ključne riječi: Internet of things, Interoperabilnost, IoT uređaji, Promet, Senzori

Sadržaj

Sadržaj iii
1. Uvod 1
2. Domena, metode i tehnike rada 2
3. Značaj interoperabilnost i IoT-a 3
3.1. Teorijska osnova IoT-a 3
3.2. Tehnologije 4
3.2.1. RFID 5
3.2.2. WSN 6
3.2.3. Middleware 7
3.2.4. Cloud Computing 8
3.2.5. Iot application software 9
3.3. Arhitektura IoT-a 10
3.4. Izazovi 11
3.4.1. Skladištenje podataka 11
3.4.2. Data mining 12
3.4.3. Privatnost 12
3.4.4. Sigurnost 12
3.4.5. Kaos 13
3.5. Primjena i povećanje vrijednosti kroz IoT 14
3.5.1. Osobna primjena 14
3.5.2. Primjena u realnom sektoru 14
3.5.3. Primjena u javnom sektoru 15
3.5.4. Mobilna primjena 16
3.6. Interoperabilnost 16
4. IoT model prometnog sustava 18
4.1. Sastavnice modela 18
4.1.1. Funkcionalnosti i programski kod 19
4.1.2. Pseudo kodovi važnijih elemenata simulacije 25
4.2. Prikaz rada simulacije 27
4.3. Nadogradnje sustava 31
5. Zaključak 34
Popis literature 35
Popis slika 36

Popis tablica 37
Prilozi (1, 2, …)..... 38

1. Uvod

Internet stvari ili skraćeno IoT jedan je od najpoznatijih trendova u IKT-u 21. stoljeća. Iako ne možemo sa sigurnošću označiti točan datum i mjesto nastanka fraze Internet of things, prema više izvora moguće je zaključiti da je ona skovana u SAD-u između 1999. i 2008. godine. Pogled na Internet of things s vremenom se mijenjao, a danas on predstavlja više uređaja u određenoj komunikaciji, koji razmjenjuju poruke i tako međusobno utječu na ponašanja drugi uređaja u grupi.

Interoperabilnost je jedna od osnovnih značajki Interneta stvari. To je svojstvo ostvarenja komunikacije, odnosno razmjene podataka. Bez mogućnosti da dva uređaja komuniciraju oni ni ne mogu biti dio IoT networka.

Prema Gartner(2014) procjena je da je IoT 2009. godine uključivao oko 0.9 milijardi uređaja, a u 2020. taj broj će premašiti 26 milijardi uređaja. Važnost IoT-a je neupitna, a njegova sveprisutnost postaje očigledna na svakom koraku. Težnja praktičnog dijela ovog rada je prikazati kako preinaka određenog sustava iz svoj klasičnog stanja, u kojem njegovi dijelovi međusobno ne ostvaruju komunikaciju, na sustav koji funkcionira po IoT standardima, može uvelike doprinijeti efikasnosti sustava ili čak stvoriti nove efekte koje sustav ostvaruje na okolinu.

2. Domena, metode i tehnike rada

Kako se IoT u proteklim godinama rapidno širi, te njegova primjena postaje gotovo sveprisutna, gotovo je nemoguće u potpunosti ga opisati. Domena opisana ovim radom sastoji je od glavnih obilježja IoT-a bitnih za razumijevanje značaja IoT-a, njegovih početaka i svrhe. Nadalje tu osnovnu domenu rad proširuje na ono najbitnije za razumijevanje rasta i primjene IoT-a, kroz detaljnije opise osnovnih tehnologija IoT-a, njegove arhitekture, izazova, te završno primjene.

Kao nastavak, odnosno proširenje domene primjena IoT-a, u rad je uključen i praktični primjer izrade IoT sustava. Primjer se odnosi na više uređaja koji tvore prometni sustav, a imaju obilježja interoperabilnosti, visoku razinu međusobne komunikacije, međusobno uvjetovanje djelovanja, te ostale značajke IoT-a.

Za proučavanje teorijske strane IoT-a, te stvaranje pregleda i prikaza IoT-a kroz vrijeme, u odnosu na druge tehnologije i u odnosu na čovjeka, korištene su metode prikupljanja različitih izvora istraživanjem literature – knjiga, članaka, Internet izvora.

Za izradu praktičnog dijela, odnosno primjera IoT sutava korišteni su ponajviše Internet izvori – kao priprema, učenje, a tehnike koje su korištene za samu izradu uključuju: programiranje unutar .net razvojnih okolina, kreiranje simulacije i proučavanje implementacija stvarnih sustava sličnih simuliranom.

3. Značaj interoperabilnost i IoT-a

Ovo je glavni dio rada u kojem treba razraditi temu, pojasniti istraživanja, prikazati rezultate i slično. Poželjno je na početku poglavlja dati kratki opis strukture poglavlja, kako bi čitatelj/čitateljica rada mogao/mogla lakše pratiti složenu cjelinu.

3.1. Teorijska osnova IoT-a

Prema Lee, Lee() IoT je paradigma zamišljena kao globalna mreža uređaja i strojeva s mogućnošću međusobne komunikacije. Autori ovu definiciju nadalje proširuju na sustav u kojem uređaji i strojevi funkcioniraju, odnosno kojem oni pridodaju vrijednost. Kad uređaji i strojevi zbog međusobne komunikacije ostvaruju bolje rezultate u npr. vođenju zaliha, podršci korisniku, proizvodnji, poslovnoj inteligenciji ili analitici, tada govorimo o uspješno realiziranom IoT sustavu. U domeni produkcije, skladištenja i prodaje proizvoda ulaganja u IoT donose izrazito velika poboljšanja u praćenju i nadzoru materijala, proizvoda i poluproizvoda. Ulaganja u IoT zahtijevaju promjene u tijeku rada, što dovodi do optimizacije poslovanja i smanjenja troškova i korištenih materijala i sirovina.

Internet stvari moguće je realizirati unutar triju paradigmi, ovisno o dijelu sustava na koji je odabrana realizacija fokusirana: senzori (things), semantika (knowledge) ili Internet (middleware).

Prema Chen et al.(2014) IoT mora imati sljedeće karakteristike:

- Sveobuhvatna percepcija
  Koristeći senzore, barkodove i druge načine prikupljanja informacija i komuniciranja, IoT se uklapa u okoliš čovjeka, a njegova sposobnost da prepoznaje svijet oko sebe, prisutne objekte i lokacije, temelj je uvođenja sveobuhvatnog IoT-a.
- Pouzdan prijenos podataka
  IoT kreira interakcije koje do sada nisu postojale unutar fizičkog svijeta, virtualnog i digitalnog svijeta, te međudruštveno. Machine-to-machine komunikaciju autori navode kao osnovu za stvaranje tehnologija Network of Thing koja predstavlja veze i komunikacije Human-to-Machine, Machine-to-Machine i Mobile-To-Machine.
- Inteligentno procesiranje
  S obzirom na milijarde poruka prisutnih unutar IoT sustava, autori navode cloud computing kao tehnologiju neophodnu za IoT.

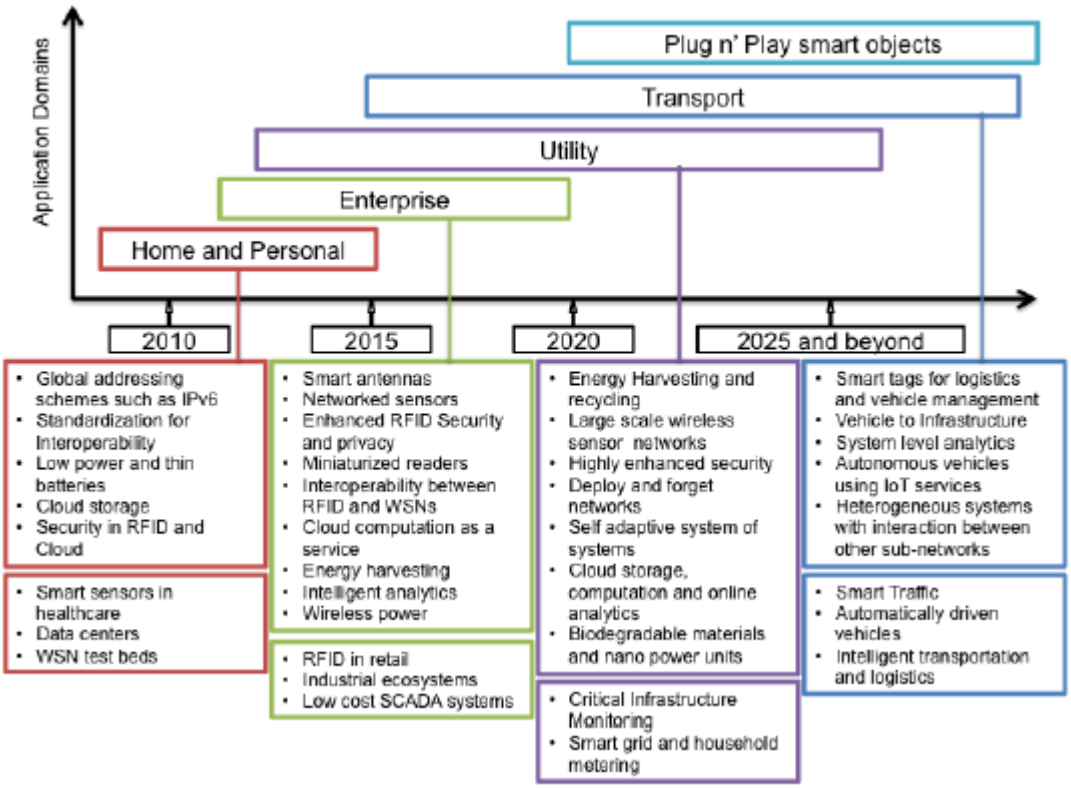
3.2. Tehnologije

U svaku od paradigmi interneta stvari možemo uvrstiti desetke tehnologija, sljedeći naslovi prikazuju one najbitnije grupe. Evoluciju IoT tehnologija možemo promatrati kroz sljedeću tablicu.

	Do 2010.	2010-2015	2015-2020	Nakon 2020
Network	Senzori	Samosvjesna i samo-organizirajuća mreža	Mreža svjesna svojeg konteksta	Mreža koja razmišlja (self-learning, self-repairing)
Software i Algoritmi	Relacijske baze podataka IoT orijentirani sustavi baza podataka Platforme temeljene na događajima Senzorski network i middleware Jednostavni lokalni algoritmi	Skalirajući softver Interoperabilni algoritmi Društveni i Poslovni sustavi aplikacija	Orijentacija na postizanje ciljeva Distribucija inteligencije za brže i kvalitetnije rješavanje problema Things-To-Things kolaboracije	User-oriented sustavi Things-To-Humans sustavi IoT 4 All Nevidljivi IoT
Hardver	RFID hardver Senzori ugrađeni u mobilne uređaje NFC hardver MEM (Micro-electro-mechanical) tehnologija	Multiprotocol i multistandard čitači Rapidni rast broja senzora Povećanje sigurnosti	Pametni senzori (biomehanika) Veći broj kompaktnijih senzora	Nanotehnologija i moderni materijali
Obrada podataka	Podaci sa seral portova Paralelan obrada podataka Naglasak na Quality of services	Naglasak na kontekstu podataka, frekvenciji pojavljivanja unutar domene i potrošnji energije	Obrada unutar konteksta s ciljem omogućavanja generiranja odgovora	Kognitivna obrada i optimizacija

Tablica 1: Evolucija IoT-a (prema Sundmaeker et al (2010))

Sljedeća slika prikazuje razvoj IoT tehnologija i proširenje primjene, kroz vremenski period od 2010. godine nadalje, uz predviđanje budućnosti. Počeci IoT-a nastaju u osobnoj upotrebi, kroz želju pojedinca da si olakša svakodnevnicu i vlastitim idejama stvori određeni IoT sustav. Oko 2015. godine IoT postaje sve više podrška poduzećima u vidu organizacije toka roba i zaliha, kroz ERP sustave, kroz online povezivanje i slično. Do 2020 i nakon očekivanje je da IoT uđe u javni sektor (bolnice, energetika, promet) i postane stalna podrška javnom sektoru. Krajnja razina tehnologija koju je sada moguće predviđati su „Plug and play smart objekti“, a ona se odnosi na plug-and-play povezivanje inteligentnih i autonomnih vozila u prometu, u logistici, u autonomnosti sustava prijevoza opće.



Slika 1:Razvoj IoT tehnologija od 2010. do danas (izvor: Gubbi et al 2013)

3.2.1. RFID

RFID je tehnologija zaslužna za uzlet bežične komunikacije koja omogućuje vrlo lako međusobno prepoznavanje uređaja. Pasivni RFID tagovi za obavljanje funkcionalnosti ne zahtijevaju električnu energiju, već koriste energiju čitača, što dopušta široku i efikasnu primjenu u mnogim područjima podržanima IoT-om, kao što su skladištenja robe ili različite vrste transporta. Aktivni RFID tagovi omogućuju iniciranje komunikacije, a ponajviše se koriste u nadziranju pošiljaka u cargo prijevozu.

Glavni razlog korištenja RFID-a u IoT sustavima je jednostavno ostvarenje identifikacije. Iako po korištenju nije nimalo složenije od očitavanja bar kodova, RFID očitavanje omogućuje pristup većem broju podataka. Oznaka RFID-a sadrži podatke u obliku Electronic Product Code-a koji se sastoji od 4 dijela. To su header, EPC manager, object class i serial number, kao što je prikazano primjerom na slici **broj slike**. Header sadrži broj verzije EPC-a, pomoću broja verzije čitač prepoznaje u kojem formatu je provedeni tag. EPC manager je globalno jedinstveni broj pripisan kompaniji proizvođaču proizvoda koji je označen tag-om, a pripisuje ga EPCglobal. EPC global je organizacija koja nastoji uspostaviti adaptaciju i standardizaciju EPC tehnologije na svjetskoj razini. Treći dio EPC oznake, object class, je jedinstveni broj koji označava proizvodnu kategoriju, a četvrti dio oznake je jedinstveni serijski broj pripisan svakom proizvodu od strane proizvođača. U tablici broj 2 prikazane su veličine određenog dijela taga u bitovima.

21	203D2A9	16E8B8	719BA30C3
Header	EPC Manager	Object Class	Serial Number
8 bitova	8 do 35 bitova	39 do 56 bitova	60 do 95 bitova

Tablica 2: Veličine dijelova EPC taga (izvor: https://www.researchgate.net/figure/The-Electronic-Product-Code-EPC-example-of-a-96-bit-EPC-tag-Adapted-from-Leong-et_fig3_28149608)

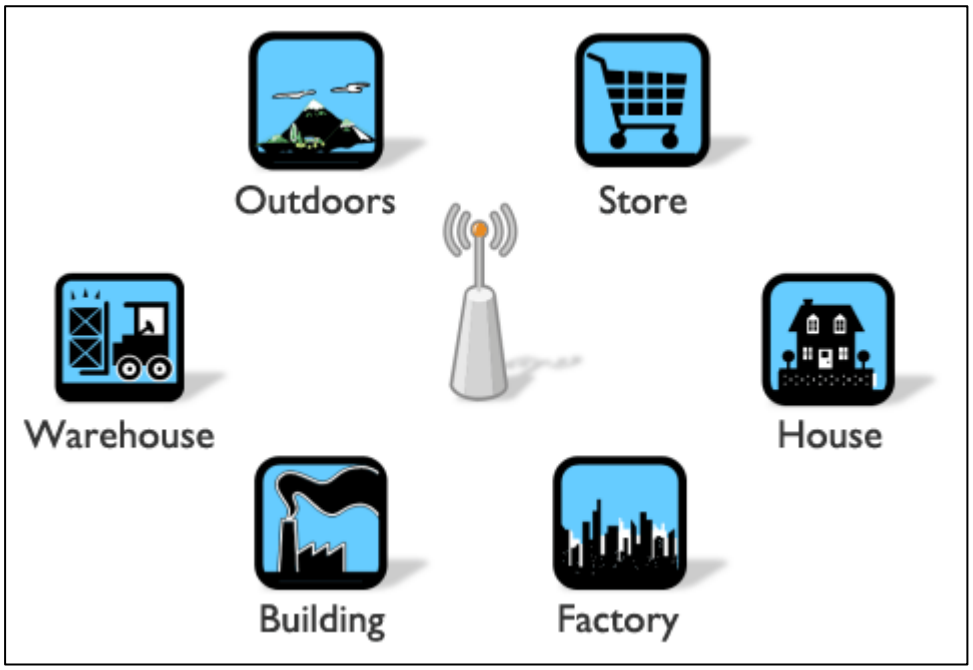
3.2.2. WSN

Wireless sensor network je grupa tehnologija koje se razvijaju u proteklih nekoliko godina, a predstavljaju jednostavne, efikasne i jeftine elektroničke sklopove sa senzorima koji obavljaju određenu vrstu mjerenja (sensoring), te imaju mogućnost ostvarivanja komunikacije, a ne iziskuju snažno strujno napajanje. U svojim počecima WSN se koristio u dijelovima logističkog lanca, međutim s padom cijena njegovih komponenti, razvojem programiranja i povećom dostupnosti znanja, njegove primjene postaju sveprisutne – od pametnih automobila i pametnih domova, do poljoprivrede, istraživanja prirode i mnogo drugih područja.

Jedan od osnovnih principa korištenja IoT-a unutar industrijskih i logističkih sustava je povezivanje WSN i RFID tehnologija u svrhu praćenja artikala kroz više kontrolnih točaka, pri čemu određeni artikl nosi RFID tag, a svaka točka sadrži senzor za očitavanje taga te po potrebi dodatne senzore.

Primjena WSN-a je izrazito široka, npr. koristi se u energetskom sektoru za real-time analizu podataka vezanih za proizvodnju električne energije unutar vodenih i vjetro-elektrana, te njihova potrebna održavanja. WSN također koriste i avioprijevoznici za sakupljanje podataka o letu, kako bi osigurali pravovremeno servisiranje aviona. Sljedeća slika prikazuje različite

elemente koje je moguće spojiti u WSN. Slika je preuzeta s stranice jednog od proizvođača senzora koji se koriste u WSN sustavima.



Slika 2: Elementi WSN sustava (izvor: http://www.ginsei-jp.com/wsn.html)

Prema Lee, Lee (2015) avioprijevoznička kompanija American Airlines koristi senzora dovoljno da za svaki let zabilježi 30 terabajta podataka. Ti podaci koriste se za preventivne servise i uvelike podižu sigurnost putovanja.

3.2.3. Middleware

Middleware je softverski sloj koji se nalazi između aplikacija prezentacijskog sloja i elektroničkog sloja koji čine WSN, RFID tehnologije i ostali input/output uređaji. On omogućuje aplikacijama prezentacijskog sloja da imaju pristup podacima potrebnima za rad i ostvarenje komunikacije, a kao jedna od najbitnijih osobina Middleware-a navodi se i skrivanje nepotrebnih podataka koji dolaze iz nižih slojeva, međutim Middleware ih ne prenosi višim.

Middleware je postao iznimno popularan u 1980-ima kada je iskorišten da pojednostavi implementaciju legacy tehnologija u nove tehnologije, upravo zbog toga jer je filtrirao velike količine podataka iz kompleksnog sustava, te dostavljao samo ono potrebno sustavu iznad sebe.

Važnost middleware-a leži u potrebi za razvojem velikog broja novih IoT sustava. Kako bi razvoj takvog sustava bio što kvalitetniji, brži i jednostavniji, koriste se middleware slojevi na koje se nadovezuje novi softver, bez potrebe da se ponavlja razvoj od najnižih razina: senzori, elektronički sklopovi, komunikacije.

3.2.4. Cloud Computing

U današnje vrijeme Cloud je tehnologija čija primjena se širi u sve grane informatike, bilo za privatnu ili poslovnu upotrebu. Posebno je korisna u IoT-u jer omogućuje backend rješenja za spremanje i obradu izrazito velikih količina podataka koji se generiraju na uređajima IoT-a.

Prema Jayavardhana Gubbi,a Rajkumar Buyya,b* Slaven Marusic,aMarimuthu Palaniswamia možemo razlikovati dvije perspektive IoT-a: Internet-centričnu i thing-centričnu, odnosno orijentiranu na uređaje.

Prema Hwhang, Chen unutar cloud computinga povezanoga s IoT-om posebno je bitno osigurati kvalitetne podatke koje sustav zadržava. To se provodi sljedećim operacijama:

- Data mining – otkrivanje, skupljanje, agregiranje, transformacija i procesiranje velikih datasetova. Ovo je fundamentalna operacija za svaki big dana sustav, a služi kao osnova za otkrivanje znanja u podacima. Mining se odnosi na sve dostupne vrste podataka, kako numeričke i tekstualne, tako i na slikovne zapise, videozapise, uzorke ponašanja i ostale dostupne podatke. Autori smatraju mining (odnosno njegovu podoperaciju loading) najkompleksnijim dijelom cjelokupnog procesa zbog potrebe da izlaz operacije čini što više podataka, ali uz očuvanje kvalitete i integriteta.
- Data Aggregation and Integration – pretprocesiranje podataka kako bi se dostigla veća kvaliteta podataka. Cilj operacija ove cjeline je čišćenje podataka, brisanje redundantnih podataka, provjeravanje relevantnosti, redukcija nepotrebnog, transformacija podataka i postizanje diskretizacije podataka. Diskretizacija podataaka je proces pretvorbe kontinuiranog skupa podataka u skup podataka s definiranim konačnim brojem intervala unutar kojih se ti podaci mogu smjestiti.
- Machine Learning i Big Dana Analytics – procesi koji iskorištavaju potpunu snagu cloud-a kako bi analizirali velike datasetove znanstvenim i statističkim metodama. Pomoću kompleksnih računalnih programa sustavi automatski prepoznaju kompleksne uzorke unutar podataka i temeljem stečenog znanja korisnik omogućuju podršku u donošenju odluka.

U usporedbi s tradicionalnim datasetovima, big data sadrži mnogo više podataka. Ti podaci su nestrukturirani i zahtijevaju real-time analize, a kao rezultat obrade prije pokazanim operacijama iz njih se otkrivaju nova znanja. Primjene novostečenih znanja su različita ovisno o sustavu, ponajviše su to razumijevanje skrivenih vrijednosti, povećanje efektivnosti i efikasnosti organizacije i upravljanja podacima i resursima i slično.

Razlog izrazito brzog rasta broja big data sustava leži u sve većoj povezanosti svakodnevnog života ljudi sa internetom, web servisima i cloud sustavima.

3.2.5. Iot application software

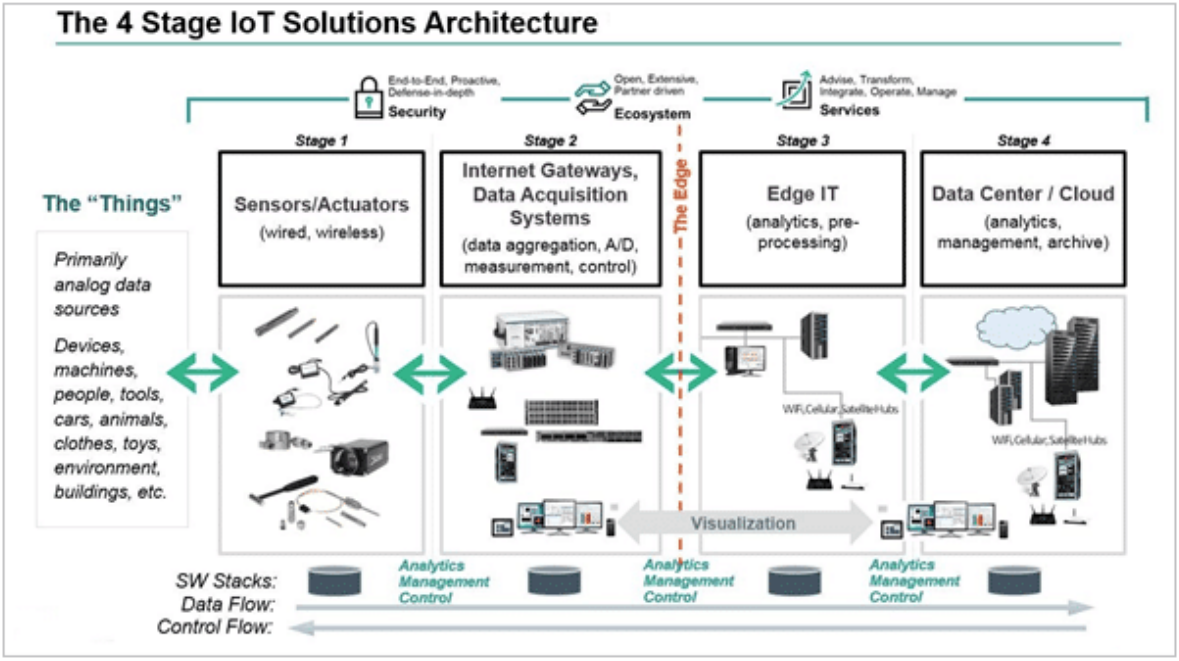
Kod razvoja aplikacija unutar IoT-a naglasak je na korisniku. Kako middleware omogućuje povezivanje senzora i aplikacijskog sloja, tako aplikacijski sloj omogućuje da se čovjek poveže s sustavom. Aplikacijski sloj osigurava da korisnik u zadanom vremenskom roku vidi bitne informacije, te djeluje na temelju viđenog. Za razliku od komunikacije između uređaja, kod komunikacije sa čovjekom potrebno je razviti i neku vrstu vizualizacije, te podatke s uređaja preoblikovati u jednostavan i čitljiv oblik.

Težnja IoT sustava mora biti razvoj dovoljne razine inteligencije da čovjek nije ni potreban sustavu, već kroz identifikaciju problema i međusobnu komunikaciju uređaji sami rješavaju novonastale situacije.

3.3. Arhitektura IoT-a

Arhitektura IoT-a razvija se rapidno sa sve većom primjenom IoT-a i razvojem samih „stvari“. Sljedeća slika prikazuje kompleksnost modernog IoT-a, ali i zašto je on bitan. Slika prikazuje arhitekturu kroz 4 razine. Nulta razina modela, na slici sasvim lijevo, odnosi se na ljude, životinje, okolinu sustava općenito, jednostavne uređaje (primitivnije od prve razine), zgrade i slično. Prva razina fizički input/output elektronički mehanizam: senzor, aktuator, motor, displej i njihove kombinacije. Ovo je rubna razina koja sadrži uređaje koji sami po sebi, bez programske podrške, ne ostvaruju neku funkcionalnost ili ostvaruju tek minimalnu; npr. jednostavni mehanizam s kamerom snima sliku svakih 3 sekunde i sprema ju na sd karticu. Međutim nema kolanja podataka van razine. Na drugoj razini nalaze se sustavi za prometovanje podataka, skladištenje i pristup internetu. To čini mrežna oprema kao što su modemi, ruteri, mrežne kartice, antene te računala koja barataju podacima. Prva i druga razina generiraju Dana Flow i proslijeđuju ga trećoj i četvrtoj razini. Treća razina sustava naziva se *Edge IT*, a orijentirana je na analizu i pretprocesiranje podataka, sastoji se od servera i snažnijih računala koji obrađuju podatke funkcijama kao što su filtriranja i sortiranja podataka. Četvrta razina arhitekture služi za analizu, upravljanje, arhiviranje i stvaranje vizualizacije podataka. Ovo je programski najrazvijenija razina, ona sadrži najveću količinu podataka (sprema sve bitne podatke), a služi za stvaranje kompleksnih izvještaja, ostvarenje umjetne inteligencije i skladištenje podataka.

Cjelokupna arhitektura IoT-a mora biti zaštićena sigurnosnim mehanizmima, otvorena prema nadogradnjama i poveznivanjima s drugim arhitekturama i usmjerena ka informiranju, usmjeravanju i podršci operacijama poslovnog ili nekog drugog sustava unutar kojega je IoT realiziran. Dva su bitna toka koji nastaju u IoT arhitekturi, a to su tok podataka i kontrolni tok (Data Flow i Control Flow). Tok podataka kreće od nižih razina do viših i rezultira agregiranjem velike količine podataka s senzora i mehanizama prve razine. Kontrolni tok kreće od viših razina po principu povratne veze na podatkovni tok i završava uputama za uređaje na prvoj i drugoj razini ili korisnike/okolinu s nulte razine.



Slika 3: Faze različitih rješenja IoT arhitektura (izvor: https://www.researchgate.net/figure/The-4-Stage-IoT-Solutions-Architecture_fig12_329269702)

3.4. Izazovi

Izazovi IoT-a uglavnom proizlaze iz načina na koji se IoT razvija. Što je više prepoznat potencijal IoT-a i njegov doprinos sustavima, to više raste i potražnja za IoT rješenjima. Za izrazito veliku potražnju nastoje se ponuditi što brža rješenja pri čemu je naglasak na funkcionalnosti sustava za korisnika, a dijelovi sustava koji podržavaju te funkcionalnosti ostaju zanemareni. Problemi također nastaju zbog razvoja cyber kriminala, ali i zbog gomilanja enormnih količina podataka koje je potrebno skladištiti i analizirati.

3.4.1. Skladištenje podataka

Jedan od najvećih izazova IoT-a u njegovim počecima bilo je skladištenje podataka. Veliki broj senzora i uređaja unutar IoT-a generiraju velike količine podataka koje treba kvalitetno procesirati i na siguran i pregledan način uskladištiti. Tada dostupne arhitekture podatkovnih centara nisu dostajale zadacima pred njima, a pa se pojavila potreba za novim rješenjima. Podatkovni centri postaju distribuirani kako bi se povećala efikasnost pristupa podacima i smanjilo vrijeme odgovora na upite. Prema Gubbi et al.(2013) važno je razvijanje algoritama umjetne inteligencije koji mogu biti centralizirani ili distribuirani prema potrebi, u svrhu davanja smisla skladištenju kolekcija podataka na određeni način. Osim skladištenja podataka, autori kao probleme koji se pojavljuju navode i vlasništvo nad podacima te rok trajanja podataka.

3.4.2. Data mining

Data mining alati su specijalizirani alati za procesiranje i analizu velikog broja podataka. S obzirom na veliku količinu podataka koji stiže u IoT sustav od njegovih krajnjih elemenata vezanih za WSN, potrebno je osigurati kvalitetno filtriranje i obradu tih podataka. Sami podaci se odnose na historijske zapise, ali također i na podatke iz live-streama. Data mining koristi se za obradu obadvije vrste podataka, kroz korektivne procese koji prepoznaju koji podaci kako utječu na procese unutar sustava i sugeriraju koje su nužne operacije za ostvarivanje drugačijih rezultata na senzorima (postizanje određenih željenih rezultata).

Podaci koje analiziraju stručnjaci unutar Data mininga oni moraju razumjeti, kako u smislu njihovih vrijednosti tako i u njihovom kontekstu, s ciljem da na njih mogu primijeniti prigodne matematičke i računalne metode. Posao analitičara svodi se na iskorištavanje Big Data skupova podataka za donošenje poslovnih odluka. Potrebe za analitičarima rastu s razvojem IoT-a, a trenutno samo u SAD-u postoji manjak od nekoliko stotina tisuća analitičara.

3.4.3. Privatnost

Kako je već više puta navedeno u tekstu rada, IoT sustavi generiraju ogromne količine podataka. Velika većina tih podataka na neki se način odnosi na ljude i koristi se za povećanje kvalitete njihovih života. Međutim rapidan razvoj IoT-a nerijetko ne prati i održavanje privatnosti podataka koji nastaju ili kolaju unutar IoT-a. Iako su težnje da se privatnost održi što je moguće većom, odnosno da se pronađe kvalitetan balans između privatnosti i podatka potrebnih sustavu, mišljenja ljudi o privatnosti unutar IoT-a nisu sasvim pozitivna. Prema Lee(2015) u 2014 godini, prema TRUSTe IoT Privacy Indeks-u samo 22% korisnika interneta izjasnilo da smatra napredak IoT-a bitnijim od smanjenja privatnosti.

3.4.4. Sigurnost

Prema istraživanju Hewlett Packarda iz 2014. godine otkriveno je da čak 70% IoT uređaja sadrži ozbiljne slabosti u svom sigurnosnom sustavu. U prosjeku je svaki uređaj sadržavao 25 slabih točaka koji bi se mogli iskoristiti na štetu uređaja, sustava ili mreže na koju je povezan.

Veliki dio IoT sustava izrađen je na infrastrukturama sa slabim ili gotovo nikakvim sigurnosnim strukturama, a česti je slučaj i da IoT sustavi prikupljaju mnogo podataka osobnim i osjetljivih podataka o svojim korisnicima, njihovim domovima ili drugim nekretninama/pokretninama, zdravlju, financijama i slično.

Saznanja o slaboj zaštiti podataka i privatnosti unutar IoT-a već sada je počela korisnike odmicati od implementacija i korištenja IoT sustava, što možemo gledati i s pozitivne

strane, jer takav trend tjera proizvođače i developere IoT-a da sve više fokus stavljaju na takve kritične aspekte sustava koji razvijaju.

3.4.5.Kaos

Kaos je pojam sa teškim prizvukom i gotovo ga je nemoguće zamisliti kako opisuje okruženje toliko povezano s IKT-om. Međutim IoT je trend čiji se ciklus razvoja u potpunosti razlikuje od uobičajenog ciklusa proizvoda dostupnih unutar IKT-a i šire. IoT raste neuobičajenom brzinom i to već dugi niz godina, a raširenost njegove primjene ne stignu pratiti razvoj potrebnog sigurnosnog sustava, razvoj standardizacije i normi. Osjetljivost manjeg dijela sustava ili njegova ranjivost ne moraju nužno biti kritični kada govorimo o hermetičkom sustavu, međutim u otvorenom sustavu, povezanom na Internet, koji surađuje sa velikim broj drugih sustava svaka stavka morala bi imati kvalitetne sigurnosne mehanizme. Kod IoT-a to nije slučaj. Sustavi se nadovezuju jedan na drugi, a njihove komponente fokusirane su na generiranje podatka, prijenos i skladištenje, dok sigurnost ostaje zanemarena, koliko zbog ne znanja toliko i zbog nemara i težnje da se stvori što više u što kraćem vremenu. Takvi sigurnosni propusti možda nikada neće doći na naplatu, međutim ako dođu to može uzrokovati kolaps sustava koji se poput domina širi od jednog slabo zaštićenog dijela prema drugima.

Ako za primjer uzmemo sustav pametnog doma („smart home“) koji povezuje veći broj senzora unutar manjih uređaja i komponenti, a odgovoran je za krucijalne sigurnosne elemente kao što su vrata kuće ili garaže, protupožarni sustavi, povezivanja računala i ostale informatičke opreme u jedinstvenu mrežu, potrebno je pažljivo upravljati svaki njegovim dijelom. Stavimo npr. u takav sustav loši termostat. Jednostavnu napravu s zanemarivom cijenom i krajnje malim brojem funkcionalnosti. Napadač uspijeva lako zavarati termostat i on šalje signal sustavu da je nastupio požar. Vrata kuće se otvaraju kako bi ukućanima omogućila brzi i siguran bijeg od vatre. Napadač ulazi u kući. Jesu li ovakvi propusti nemogući? Ne. Štoviše njihova jednostavnost je upravo ono što dovodi do zanemarivanja njihove stvarne važnosti. Ovaka problem vrlo lako bi bio riješen postavljanjem više senzora koji bi kombinirano mogli ocijeniti da li je opasnost stvarna, odnosno u stanju nedoumice javiti podatke korisniku, povezati ga s kamerama ili drugim mehanizmima za provjeru stanja.

Moguće je lako doći do zaključka da je svaki i najmanji dio IoT sustava potrebno sigurnosno zaštititi ponajviše u svrhu očuvanja privatnosti i osiguranja vjerodostojnosti podataka koji izlaze iz takvog manjeg uređaja prema sustavu.

3.5. Primjena i povećanje vrijednosti kroz IoT

Sljedeći odlomak opisuje primjenu IoT-a kroz 4 domene koje se odnose na njegove korisnike. Osim ovog pristupa, primjenu IoT-a možemo kategorizirati na mnogo načina, npr. kroz geografsku pokrivenost, kroz mrežu kojom se provodi ili pak kroz uključenost korisnika. Ovakva podjela najbolje će poslužiti kako bi se određenoj domeni pridružilo povećanje vrijednosti koje IoT donosi za njezine učesnike.

Prema Lee, Lee (2015) evaluaciju ulaganja u IoT moguće je provesti na dva načina: računanjem neto sadašnje vrijednosti i kroz „real options“ pristup. „Real options“ pristup svodi se na vrednovanje različitih mogućnosti koje postoje za poduzeće ili osobu putem nekog mehanizma višekriterijskog odlučivanja – npr. kroz stablo odluke u kojem promatramo trošak, ali i dobit i slično. Autori preporučaju odlučivanje s kombinacijama više opcija zbog prirode IoT tehnologija (njihovog brzog razvoja, promjenjivosti i prisutnog rizika. Kroz klasični izračun NPV-a također je moguće računati vrijednosti projekta uz određene promjene u okolini, međutim taj je pristup orijentiran samo na financijsku stranu ulaganja, dok se u „real options“ pristupu može kombinirati financije s drugim aspektima ulaganja.

Navedene domene često se preklapaju, odnosno sustavi i/ili korisnici mogu se istovremeno nalaziti u više domena ili često prelaziti iz jedne domene u drugu.

3.5.1. Osobna primjena

Domena osobne primjene IoT-a je ona domena u kojoj su podaci skupljeni unutar IoT sustava dostupni samo vlasniku mreže na kojoj se sustav nalazi, pri čemu je osnova IoT komunikacije unutar mreže najčešće WiFi. U ovakav IoT sustav često je uključen i mobilni telefon korisnika koji na određeni način služi kao prezentacijski sloj i komunikacijski sloj putem kojeg korisnik upravlja dijelovima sustava. Gubbi et al.(2013) navodi IoT unutar osobne domene primjene kao savršenu platformu za postavljanje sustava sveprisutnog zdravstva. U tom sustavu različiti uređaji prate fiziološke parametre osobe, te putem bluetootha prenose informacije do mobilnih uređaja ili drugih sučelja koja omogućuju neku vrstu praćenja i nadzora životnih funkcija. Razvoj ovakvih sustava nadalje vodi prema razvoju kućnog nadzora (brige) o starijim i nemoćnim, čime se smanjuju potrebe za hospitalizacijama i preseljenjima u staračke domove.

3.5.2. Primjena u realnom sektoru

Slično definiciji domene osobne primjene, primjena u realnom sektoru definirana je kao skup IoT sustava koji generiraju podatke koji se koriste unutar određene kompanije. Počeci

korištenja IoTa u realnom sektoru bili su sustavi za praćenje i upravljanje dobrima korištenima u proizvodnji i logistiki poduzeća.

Senzori su već duže vremena osnova mnogih tvorničkih postrojenja, kao dio strojeva za proizvodnju i preradu, kao dio sigurnosnog ili automatizacijskog sustava, sustava klimatizacija i slično. Navedeni senzori i jednostavni elektronički sklopovi s vremenom su se povezivali putem mreže i tako stvarali početne IoT sustave.

Najjednostavniji sustavi unutar realnog sektora su Smart Office sustavi s manjim brojem korisnika, koji koriste lokalne servere, manji broj uređaja i uglavnom komuniciraju WiFi-om, 3G ili 4G mrežom. Srednje veliki sustavi uključuju Smart Retail i Smart Agriculture sustave s većim brojem korisnika, čiji se podaci često skladište i na dijeljenim serverima, u odnosu na lokalne kod manjih sustava. Najveći sustavi IoT-a su sustavi za transport, upravljanje energentima kao što su voda, nafta i električna energija, a u odnosu na prethodno navedene sustave ovdje je prisutna satelitska komunikacija, veći broj dijeljenih servera, veliki broj korisnika i mnogo veći broj senzora i uređaja.

3.5.3. Primjena u javnom sektoru

Najčešće primjene IoT-a u javnom sektoru su domene sigurnosnog nadzora, nadzora kritičnih infrastruktura, monitoringa okoliša, monitoringa zdravstvenih sustava i transporta. Prema Gubbi et al.(2010.) najveći potencijal IoT-a za primjenu u javnom sektoru leži u mogućnostima optimizacija sustava i primjenama pametnih mjerenja, odnosno kompleksnih matematičkih metoda nad velikim brojem podataka. Primjer kompleksnog mjerenja unutar sustava je mjerenje potrošnje električne energije na svim točkama sustava električne energije unutar određene zgrade, što omogućava pregled i optimizaciju korištenja. Takvo mjerenje na razini grada može uvelike doprinijeti efikasnosti korištenja električne energije i kvalitetnom održavanju balansa opskrbe i potrošnje, uz pravovremena održavanja sustava.

Isti autori spominju i IoT temeljen na video i slikovnim podacima koji integrira sakupljanje, obradu, umrežavanje i isporuku slikovnih i video zapisa na razini velegrada koji otvara vrata za nova znanstvena istraživanja na temelju velikog broja novih dostupnih podataka. Nadzor koji se tako ostvaruje omogućuje kvalitetnu borbu s kriminalom, lakši pronalazak izgubljenih stvari, proučavanje obrazaca ponašanja, analizu događaja, predviđanja kašnjenja i slično. Nadzor prometa kroz takav sustav omogućuju pravovremene informacije o gužvama, zastojima prometa i slično, temeljem čega se u prometni sustav može implementirati kvalitetno i pravovremeno planiranje i kreiranje rasporeda.

Još jedna zanimljiva primjena masovnog IoT sustava u javnom sektoru je nadzor vodoopskrbne mreže koji pridonosi osiguranju dostupnosti pitke vode za građane ili pak vode

za različita postrojenja kojima je potrebna. Nadzor sustava pitke vode omogućuje pravovremene reakcije sa što manje gubitaka u slučaju kontaminacije vode i sličnih problema. Sustav se također može kvalitetno proširiti na nadzor kakvoće tla, u korelaciji sa nadzorom voda i kao upotpunjavanje dvaju sustava međusobnom komunikacijom. To omogućuje donošenje kvalitetnih odluka vezanih za agrikulture, poljoprivredu i slično.

Prema Chen et al (2014) mreža senzora za IoT u Kini se uvodi od 1999. Već tada su započete implementacije preteča IoT-a, a 2010. godine i u službenim dokumentima definira se strategija razvoja IoT-a.

3.5.4.Mobilna primjena

Navedene domene često se preklapaju, odnosno sustavi i/ili korisnici mogu se istovremeno nalaziti u više domena ili često prelaziti iz jedne domene u drugu. Kada govorimo o mobilnoj primjeni možemo promatrati osobnog korisnika koji se kreće i koristi postojeće IoT sustave kako bi ostvario neku radnju što brže, sigurnije ili uz postizanje određenih usputnih ciljeva (razgledavanja grada, izbjegavanje gužvi, ušteda novca).

Unutar prometnog sustava IoT nudi snažna rješenja za nadzor i planiranje ruta kretanja. Pomoću WSN-a moguće je usmjeravati vozila kako bi se smanjila opterećenja na zagušenim prometnim čvorištima, moguće je analizirati obrasce ponašanja i definirati algoritme za poboljšanja.

Bitna tehnologija unutar mobilne primjene IoT-a je bluetooth koji uređajima unutar sustava omogućuje laku komunikaciju i praćenje uređaja koji se kreće. Pomoću bluetootha moguće je u vrlo kratkom vremenu obavijestiti korisnika nekog uređaja o mogućih problemima unutar prometnog ili drugog sustava, te ga usmjeriti na sigurniju rutu ili ga navesti da riješi neki drugi problem koji se pred njim pojavio.

3.6. Interoperabilnost

Prema Lee,Lee(2015) važnost IoT-a prolazi iz njegove sposobnosti uređaja unutar njega da međusobno komuniciraju. Osnova rasta takvog sustava proizlazi iz skalabilnosti. IoT je toliko bitan trend jer je njegovo nadograđivanje jednostavno. Kada bi iz dosad razvijenih IoT sustava i uređaja oduzeli njihovu moć interoperabilnosti i ograničili ih same na sebe, oni ne bi otvarali mogućnosti za nadogradnje u veće sustave, generiranje Big Data podataka i naposljetku otkrivanje znanja u tim podacima. Zbog mogućnosti skaliranja sustava IoT se rapidno širi i ideja jednog kreatora nekog sustava, već sutra može biti iskorištena za nešto potpuno drugo od strane nekog druge razvojne institucije ili pojedinca. Veliki IoT sustavi koji prate logistike velikih poduzeća ili neke aspekte javnih sektora razvijali bi se desetljećima kada

bi njihovi korisnici morali razvijati svaki uređaj i dio sustava sami. Međutim, zbog širokog raspona dostupnih rješenja i njihove lake povezivosti, razvoj velikih sustava unutar IoT-a mnogo je pojednostavljen i vremenski manje zahtjevan.

Ono što zapravo čini interoperabilnost prema Wagner(1996) jest sposobnost dvaju softvera da ostvaruju kooperaciju usprkos razlikama u razvojnim okolinama u kojima su nastali, sučeljima i platformama na kojima djeluju. Važnost interoperabilnosti leži u skalabilnosti i mogućnosti ponovnog korištenja (engl. reusability) resursa, što se odvija tako da jedan više korisnika putem pristupnih mehanizama razmjenjuju neke resurse. Iako su definicije autora stare više od 20 godina, svakako su primjenjive i na moderni IoT u kojemu je moguće primijetiti visoku razinu interoperabilnosti. Autor dijeljenje resursa spominje između klijenta i servera, što je u današnje vrijeme evoluiralo do korištenja resursa među deviceovima koji su u takvoj informacijskoj strukturi na istoj razini, a tehnologije koje spominje kao ključne (UNIX i WWW), sada su samo osnova za desetke drugih tehnologija i tisuće dostupnih frameworkova.

Glavni mehanizmi koje Wagner (1996) spominje kao osnovice interoperabilnosti su *Interface Standardization* i *Interface Bridging*. Standardization se odnosi na mapiranje servera i klijenata u stanje u kojem su sučelja uobičajena, standardna. Bridging se odnosi na mapiranje na razini komunikacije samo servera i klijenta. Ove mehanizme možemo prepoznati i u današnjim sustavima, uz razliku da ne govorimo isključivo o klijent-server arhitekturi, već o složenijim sustavima u kojima postoje komunikacije vertikalno po istim razinama razvijenosti i funkcionalnosti uređaja i horizontalno, pri čemu uređaje/sustave viših razina možemo uspoređivati sa serverima iz vremena autora.

Bitna tehnologija unutar mobilne primjene IoT-a je bluetooth koji uređajima unutar sustava omogućuje laku komunikaciju i praćenje uređaja koji se kreće. Pomoću bluetootha moguće je u vrlo kratkom vremenu obavijestiti korisnika nekog uređaja o mogućih problemima unutar prometnog ili drugog sustava, te ga usmjeriti na sigurniju rutu ili ga navesti da riješi neki drugi problem koji se pred njim pojavio.

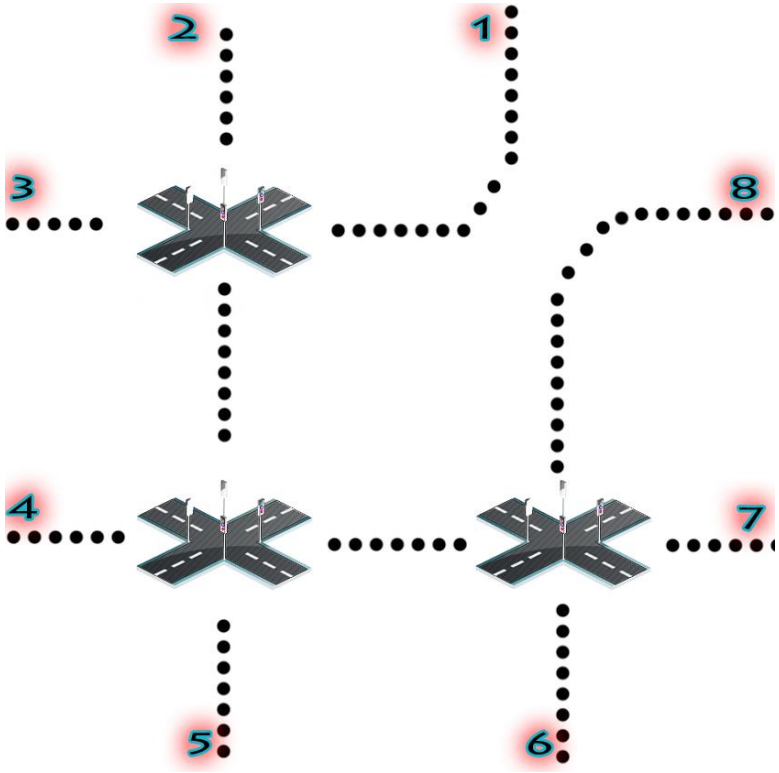
4. IoT model prometnog sustava

Prometni sustav jedan je od najučestalijih načina korištenja IoT-a unutar javnog sektora. Podržavanje prometa kroz IoT otvara mogućnost za velike uštede vremena osobnih i komercijalnih korisnika, detaljnu analizu ponašanja, povećanje sigurnosti i razvoj mnogih pametnih sustava u komunikaciji s prometnim sustavom. Sljedeće poglavlje prikazuje simulaciju korištenja IoT-a unutar modela prometnog sustava

4.1. Sastavnice modela

Model sustava sastoji se od 4 glavna dijela: 3 raskrižja i vozila hitne pomoći. Te sastavnice unutar simulacije prikazane su pomoću nekoliko odvojenih formi, kako bi se unutar programskog koda mogle prepoznati funkcionalnosti koje bi na slični način bile implementirane u stvarnom rješenju. Paralelno sa prikazom funkcionalnosti određenih elemenata sustava, biti će predočena i objašnjenja implementacija u stvarnom sustavu.

Sljedeća slika prikazuje sastavnice modela i moguće početne i krajnje pozicije vozila hitne pomoći koja prolazi sustavom. Među pozicijama se nalaze raskrižja koja su u programu simulacije označena s R1, R2 i R3, redom od prve prema posljednjoj poziciji.



Slika 4: Prikaz sastavnica modela (vlastita izrada)

4.1.1. Funkcionalnosti i programski kod

Simulacija sustava programirana je u C# programskom jeziku, a sučelje je riješeno sa 3 windows forme. Prva forma je prikaz raskrižja, prometnica i semafora, na kojima su vidljiva stanja semafora i očitavanja senzora prilikom gomilanja redova čekanja na crvenom svjetlu. Sljedeća forma prikazuje sučelje unutar vozila hitne pomoći u koje se unosi ruta kojom će vozilo proći. Kada je ruta unesena moguće je poslati poruku sustavu kako bi se otvorili potrebni semafori. Treća forma je prikaz tijeka simulacije na kojem se vidi redom koji se događaji događaju i koje su prisutne vrijednosti.

Svako raskrižje modela sastoji se od 4 semafora. Za prikaz promjena svjetala semafora u simulaciji se koriste promjene boja panela, a u realnom sustavu to bi se postiglo paljenjem žarulja unutar kućišta s različitim bojama. Svjetla semafora mijenjaju se periodično, što je pri programiranju semafora riješeno korištenjem tajmera, odnosno događaja vezanih uz tajmere: timer_tick, koji se odnosi na otkucaj danog vremenskog perioda svakog tajmera. Kako su različita raskrižja drugačije opterećena prometnim vozilima, ona imaju i drugačija vremena perioda za svjetla semafora, što je riješeno korištenjem više tajmera, odnosno svako raskrižje ima svoj tajmer. Vrijednosti tajmera, odnosno vremena trajanja određenog svijetla na semaforu su hardkodirana, međutim vrlo lako bi se mogla implementirati njihova varijabilnost. Sljedeći dio koda prikazuje rad jednog semafora (na raskrižju broj 3), njegove periodične promjene svjetla i njegov prioritetni red, kada mu prilazi vozilo hitne pomoći.

```
private void timerR3_Tick(object sender, EventArgs e)
        {
            brojacR3++;
            if (brojacR3 < 9 && citanjeZastavicaR3() == -1)
            {
                BrisanjeOznakaR3();
                semaforR3I.BackColor = Color.Red;
                semaforR3Z.BackColor = Color.Red;
                semaforR3S.BackColor = Color.LightGreen;
                semaforR3J.BackColor = Color.LightGreen;
            }
            else if (brojacR3 < 12 && citanjeZastavicaR3() == -1)
            {
                BrisanjeOznakaR3();
                semaforR3I.BackColor = Color.Yellow;
                semaforR3Z.BackColor = Color.Yellow;
                semaforR3S.BackColor = Color.Red;
                semaforR3J.BackColor = Color.Red;
            }
            else if (brojacR3 < 18 && citanjeZastavicaR3() == -1)
            {
                BrisanjeOznakaR3();
                semaforR3I.BackColor = Color.LightGreen;
                semaforR3Z.BackColor = Color.LightGreen;
                semaforR3S.BackColor = Color.Red;
                semaforR3J.BackColor = Color.Red;
```

```
}
else if (brojacR3 < 20 && citanjeZastavicaR3() == -1)
{
    BrisanjeOznakaR3();
    semaforR3I.BackColor = Color.Red;
    semaforR3Z.BackColor = Color.Red;
    semaforR3S.BackColor = Color.Yellow;
    semaforR3J.BackColor = Color.Yellow;
}
else if (brojacR3 >= 20 && citanjeZastavicaR3() == -1)
{
    brojacR3 = 0;
}
else if (citanjeZastavicaR3() > -1)
{

    IspisNaLog("Raskrižje R3 prelazi na prioritetni rad.");
    //pali sva crvena
    semaforR3I.BackColor = Color.Red;
    semaforR3Z.BackColor = Color.Red;
    semaforR3S.BackColor = Color.Red;
    semaforR3J.BackColor = Color.Red;
    //odredi od kud dolazi vozilo i upali zeleno
    if (citanjeZastavicaR3() == 0)
    {
        semaforR3Z.BackColor = Color.LightGreen;
        labelR3Z.Show();

    }
    else if (citanjeZastavicaR3() == 1)
    {
        semaforR3J.BackColor = Color.LightGreen;
        labelR3J.Show();

    }
    else if (citanjeZastavicaR3() == 2)
    {
        semaforR3I.BackColor = Color.LightGreen;
        labelR3I.Show();

    }
    else
    {
        semaforR3S.BackColor = Color.LightGreen;
        labelR3S.Show();

    }
```

Pomoći liste integera ZastaviceR3[4] raskrižje određuje ulazi ili u prioritetni rad ili ne. Ukoliko je jedna od zastavica podignuta raskrižje prema njoj toj strani otvara zeleno svjetlo, a ostalim stranama pali crvena, tako omogućujući prolazak vozilu hitne pomoći. Svako raskrižje ima svoju listu zastavica, a ona se puni iz liste Zastavice[10] koja sadrži zastavice svih pozicija. S obzirom na to da postoji više zastavica za semafore, njih ukupno 12, a pozicija je 10, prilikom punjenja zastavica program određuje kamo prolazi vozilo iz onih pozicija koja su između dvaju

raskrižja, te prema tome određuje koju zastavicu će u tom trenutku podizati. To je implementirano sljedećim dijelom koda.

```
private int citanjeZastavicaR3()
        {
            int brojZastavice = -1;

            for (int i = 8; i < 12; i++)
            {
                if (Zastavice[i] > 0) brojZastavice = i-8;
            }
            return brojZastavice;
        }
```

Zastavice se podižu unutar klase forme vozila, prema ulaznim podacima koje unosi korisnik. Sljedeći dio koda prikazuje ograničenja unosa i implementaciju unosa s forme, podaci koji se unose su početak i kraj rute vozila. Korisnik ne upisuje pozicije ručno, nego ih odabire pritiskom na gumb. Kada unese početak, korisnik ne može unijeti istu vrijednost kao kraj rute, međutim ako je pogriješio može obrisati unesenu vrijednost i unijeti drugu. Pritiskom na gumb „Započni“ korisnik pokreće generiranje rute, nakon čega vozilo započinje komunikaciju s raskrižjima. Dok vozilo putuje rutom korisniku je onemogućen unos nove rute. S obzirom da je riječ o simulaciji, nije potrebna funkcionalnost promjene rute kada prolazak prometom već traje. Kada je riječ o stvarnom sustavu i javlja se potreba za promjenom rute, rješenje bi bilo jednostavno implementirano kroz brisanje generiranih podataka o ruti i kreiranje novih. Podignute zastavice bi se spustile te bi ih zamijenile one koje odgovaraju novoj ruti. Sljedeći dio koda pokazuje funkciju kreiranje rute koja prolazi kroz matricu veličine 10 puta 10 te traži prolaz od početne do krajnje pozicije kroz maksimalno 3 raskrižja. Funkcija je u ovom slučaju riješena pomoću više petlji i uvjeta, a kod skaliranja na veći sustav bilo bi potrebno implementirati neki od matematički složenijih algoritama uz korištenje rekurzivnih funkcija.

```
public List<int> kreiranjeRute(int pocetak, int kraj)
        {
            List<int> listaSusjeda = new List<int>();
            List<int> listaCvorova = new List<int>();
            List<int> listaPozicijaMeđuRaskrižjima = new List<int>();

            int[,] rasporedPozicija = new int[10, 10] {
                {0,1,1,1,0,0,0,0,0,0},
                {1,0,1,1,0,0,0,0,0,0},
                {1,1,0,1,0,0,0,0,0,0},
                {1,1,1,0,2,2,2,0,0,0},
                {0,0,0,2,0,2,2,0,0,0},
                {0,0,0,2,2,0,2,0,0,0},
                {0,0,0,2,2,2,0,3,3,3},
                {0,0,0,0,0,0,3,0,3,3},
                {0,0,0,0,0,0,3,3,0,3},
```

```
                {0,0,0,0,0,0,3,3,3,0},
                };

            IspisNaLog("Pokretanje algoritma za kreiranje rute
vozila.");

            for (int i = 0; i < 9; i++)
            {
                int xbrojac = 0;
                for (int j = 0; j < 9; j++)
                {
                    if (rasporedPozicija[i, j] > 0) xbrojac++;
                }
                if (xbrojac > 3) listaPozicijaMeđuRaskrižjima.Add(i);
            }
            if (rasporedPozicija[pocetak, kraj] > 0)
            {
                listaCvorova.Add(rasporedPozicija[pocetak, kraj]);
                ceste.Add(kraj);
            }
            else
            {
                //ako je ruta kroz 2 čvora
                for (int i = 0; i < 10; i++)
                {
                    if (rasporedPozicija[pocetak, i] > 0)
listaSusjeda.Add(i);
                }

                foreach (int susjed in listaSusjeda)
                {
                    if (rasporedPozicija[susjed, kraj] > 0)
                    {

                        //izracun vrijednosti za zastavice ako postoji
cesta među raskrižjima
                        listaCvorova.Add(rasporedPozicija[pocetak,
susjed]);
                        listaCvorova.Add(rasporedPozicija[susjed,
kraj]);
                        ceste.Add(susjed);
                        ceste.Add(kraj);
                        break;
                    }
                }
                //ako je ruta kroz 3 čvora
                if (listaCvorova.Count == 0)
                {
                    foreach (int susjed in listaSusjeda)
                    {
                        for (int i = 0; i < 10; i++)
                        {
                            if (rasporedPozicija[susjed, i] > 0 &&
rasporedPozicija[i, kraj] > 0)
                            {

listaCvorova.Add(rasporedPozicija[pocetak, susjed]);
```

```
listaCvorova.Add(rasporedPozicija[susjed, i]);
                                listaCvorova.Add(rasporedPozicija[i,
kraj]);
                                ceste.Add(susjed);
                                ceste.Add(i);
                                ceste.Add(kraj);
                                break;
                            }


                        }
                    }
                }

            }
            return listaCvorova;
        }

        private void timerVozila_Tick(object sender, EventArgs e)
        {
            brojac++;
            if (brojac < 4)
            {
                tbTrenutnaPozicija.Text = ruta[0].ToString();
                SpuštanjeZastavica();

                if (ruta[0] == 3) PrometForm.Zastavice[ceste[0] + 2] =
1;
                else if (ruta[0] == 2) PrometForm.Zastavice[ceste[0] +
1] = 1;
                else PrometForm.Zastavice[ceste[0]] = 1;
            }
            else if (brojac < 9 && ruta.Count>1)
            {

                tbTrenutnaPozicija.Text = ruta[1].ToString();
                SpuštanjeZastavica();
                //PrometForm.Zastavice[ceste[1]] = 1;
                if (ruta[1] == 3) PrometForm.Zastavice[ceste[1] + 2] =
1;
                else if (ruta[1] == 2) PrometForm.Zastavice[ceste[1] +
1] = 1;
                else PrometForm.Zastavice[ceste[1]] = 1;
            }
            else if (brojac < 13 && ruta.Count > 2)
            {
                tbTrenutnaPozicija.Text = ruta[2].ToString();
                SpuštanjeZastavica();
                if (ruta[2] == 3) PrometForm.Zastavice[ceste[2] + 2] =
1;
                else if (ruta[2] == 2) PrometForm.Zastavice[ceste[2] +
1] = 1;
                else PrometForm.Zastavice[ceste[2]] = 1;
            }
            else
            {
                timerVozila.Stop();
```

```
                brojac = 0;
                SpuštanjeZastavica();
            MessageBox.Show("Vozilo je stiglo na odredište");
                IspisNaLog("Vozilo je stiglo na odredište \r\nSustav se
vraća u neprioritetni način rada");
                startButton.Enabled = true;
                tbRutaVozila.Clear();
                pocetakTb.Clear();
                krajTb.Clear();
                ceste.Clear();
                timerVozila.Stop();
                tbTrenutnaPozicija.Clear();
                SpuštanjeZastavica();
            }

        }
```

Unutar funkcije KreiranjeRute(x,y) vozilu se određuje ruta kroz prometni sustav. Povezanost lokacija spremljena je u matricu veličine 10*10, čiji elementi predstavljaju raskrižje između lokacija reda i lokacija stupaca. Kroz matricu program prolazi petljama dok ne otkrije preko kojih članova može stići od početne do završne lokacije.

U stvarnom sustavu za kreiranje rute kroz mapu, kao što je slučaj u GPS navigacijskim sustavima, najčešće se koristi BFS algoritam. U BFS algoritmu lokacije na mapi zamišljamo kao čvorove grafa, a njihove udaljenosti kao bridove. Takve grafove u računalo upisujemo putem matrica kako bi ih mogli obrađivati i koristiti u programu. Prema Mirshra A. (2017) GPS sustavi osim rezultat dobivenih algoritmima za pronalazak najkraćeg puta u obzir uzimaju i trajanje putovanja, ograničenja brzine, dostupne podatke o prometnim gužvama i slično. Zbog toga je česta pojava da GPS sustavi preporučaju duže rute koje prolaze autocestom, umjesto kraćih ruta putem državnih i županijskih cesta. Prema wikipedia.com u travnju 2020. godine u Zemljinoj orbiti nalazilo se ukupno 74 GPS satelita, od kojih je 31 bio u funkciji, 9 su činili rezervu, 2 su bila u fazi testiranja, 30 umirovljenih, a 2 su izgubljena nakon lansiranja.

Kada je ruta kreirana pokreće se tajmer vozila. To je tajmer koji redom prolazi kroz sve čvorove koje je program izračunao kako bi stigao od korisnikovog početnog do završnog položaja. U simulaciji je hardkodirano zadržavanje od 2 sekunde po čvoru. Kada vozilo prilazi čvoru diže se zastavica i tako zaustavlja promet na određenom raskrižju. Nakon prolaska kroz određeno raskrižje njegove zastavice se ponovo spuštaju. Zastavice se koriste kako bi preko njih dvije različite klase mogle komunicirati. S obzirom da je semafori pripadaju jednoj klasi, a ruta vozila i tajmer prolaska drugom, omogućeno je dizanje i spuštanje zastavica kojima i jedna i druga klasa mogu pristupiti.

Forma LogForm služi za ispis važnijih događaja u simulaciji. Ona je, kao i forme s raskrižjima i forma s vozilima, implementirana korištenjem tajmera. Tajmer bilježi vrijeme u milisekundama od početka simulacije i ispisuje to vrijeme uz prisutni događaj. Također koristi vrijeme kako bi prepoznao ima li novih poruka za ispis, provjeravajući ulaz svakih 1

milisekundu. Ukoliko program prepozna novu poruku on ju ispisuje na textbox iza prošlih poruka. U bilo kojem trenutku moguće je spremiti txt datoteku u kojoj se vide svi događaji iz simulacije. U sljedećem isječku koda vidljive su funkcije s forme LogForm.

```
private void button12_Click(object sender, EventArgs e)
        {
            tbLog.Clear();
        }
private void timerLogForme_Tick(object sender, EventArgs e)
        {
            brojac++;
            if (porukaStara != porukaNova)
            {
                tbLog.Text = tbLog.Text + "Sistemski brojač:" +
brojac.ToString() + " Poruka: " + porukaNova+ "\r\n";
                porukaStara = porukaNova;
            }

        }
private void saveButton_Click(object sender, EventArgs e)
        {
            File.WriteAllText("D:\\log.txt", tbLog.Text);
        }
```

Kompletnom programskom kodu moguće je pristupiti preko hiperveze sadržane pod stavkom broj 1 priloga na kraju dokumenta.

4.1.2. Pseudo kodovi važnijih elemenata simulacije

Sljedeći pseudo kodovi prikazuju tijek funkcioniranja elemenata simulacije pojednostavljeno. Prvi pseudo kod prikazuje rad jednog raskrižja, a nazivlje je prilagođeno tako da se može primijeniti i na stvarni sustav, odnosno ne na simulaciju samo s vizualnim prikazom. Raskrižje u početku rada inicijalizira zastavice i postavlja im vrijednost na 0. Tim zastavicama kasnije pristupaju drugi elementi sustava, podižu im vrijednost te tako omogućuju programu raskrižja da u pravo vrijeme gasi i pali određena svijetla. Kod svakog otkucaja tajmera provjerava se vrijednost zastavica. Ako su zastavice spuštene tijek paljenja svjetala teče po zadanim intervalima. Ukoliko je neka zastavica dignuta raskrižje propušta vozilo iz tog smjera. U stvarnom sustavu zastavice mogu podizati mehanizmi sa senzorima ili pak komunikacijski mehanizmi koji dobivaju poruke od vozila s prioritetom. U simulaciji to odrađuje druga klasa, ona koja prikazuje sučelje vozila.

```
Inicijaliziraj Zastavice[4] i postavi im vrijednost 0;
Inicijaliziraj tajmer i pokreni ga;
Na otkucaj tajmera radi
     Ako je(Sve zastavice 0 i vrijednost tajmera u prvom intervalu);
```

```
            Pali zeleno jednoj strani, pali crveno drugoj strani;
      Ako je(Sve zastavice 0 i vrijednost tajmera u drugom intervalu);
            Pali žuta svjetla na semaforima;
      Ako je(Sve zastavice 0 i vrijednost tajmera u trećem intervalu);
            Pali zeleno drugoj strani, pali crveno prvoj strani;
      Ako je(Sve zastavice nisu 0)
            Odredi iz kojeg smjera dolazi vozilo;
            Pali smjeru vozila zeleno, ostalima pali crveno;
Sve dok tajmer radi
Kraj;
```

Pseudo kod sučelja vozila odgovoran je za podizanje zastavica. U simulaciji je to riješeno pristupanjem dijeljenim varijablama. U stvarnom sustavu to bi bilo riješeno komunikacijom putem interneta, bluetootha ili nekog drugog bežičnog signaliranja. Pseudokod prikazuje redoslijed elemenata programa. Kada korisnik unese početak i kraj rute u sučelje, sustav generira rutu. Tada sustav signalizira raskrižjima redom kako prilazi, da mu je potrebno propuštanje. Prijenos rute od vozila do raskrižja također bi mogao biti implementiran odvojenim dijelom sustava koji bi zaprimio cijelu rutu, te redom „otvarao“ raskrižja prema tome gdje se vozilo nalazi, nadzirući ga putem GPS-a.

```
Inicijaliziraj početak, kraj i rutu i postavi prazne vrijednosti;
Na događaj Pritisak gumba pokreni
      Generiraj rutu;
      Onemogući unos;
      Zabilježi početak prolaska;
      Pokreni tajmer;
Kraj procedure događaja
Na otkucaj tajmera radi
      Ako je(Vrijednost tajmera u prvom intervalu)
            Digni zastavicu prvog raskrižja s rute;
      Ako je(Vrijednost tajmera u drugom intervalu i ruta sadrži više od
1 člana)
            Spusti zastavice;
            Digni zastavicu drugog raskrižja s rute;
      Ako je(Vrijednost tajmera u trećem intervalu i ruta sadrži više od
2 člana)
            Spusti zastavice;
            Digni zastavicu trećeg raskrižja s rute;
      Inače
            Spusti zastavice;
            Zabilježi kraj prolaska;
            Omogući novi unos;
Sve dok tajmer radi
Kraj;
```

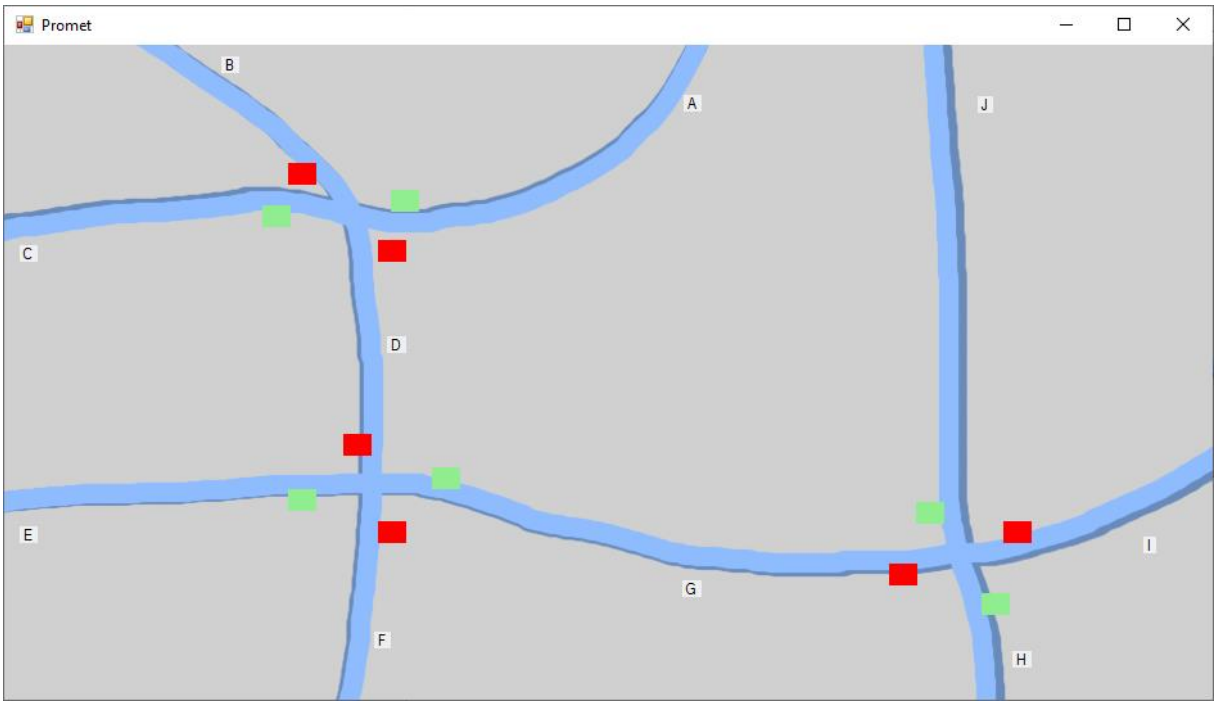
Pseudo kod forme za bilježenje događaja prikazuje dodavanje poruka na ekran kroz jednostavni mehanizam sa dva stringa: staraPoruka i novaPoruka. Ovaj mehanizam omogućuje da istom izlaznom ekranu pristupa više korisnika, bez da se gube podaci. Kada bi korisnici pokušali poslati poruke u istoj desetinki sekunde postojao bi rizik od gubitka, te bi se to dalo riješiti zaključavanjem resursa, međutim u simulaciji to nije potrebno jer je prosječni

razmak između poruka nekoliko sekundi. Kada stigne nova poruka, ona se ispisuje na ekran te postaje stara poruka, time se omogućuje usporedba u bilo kojem otkucaju tajmera. Ako se promjeni sadržaj nove poruke ponavlja se radnja.

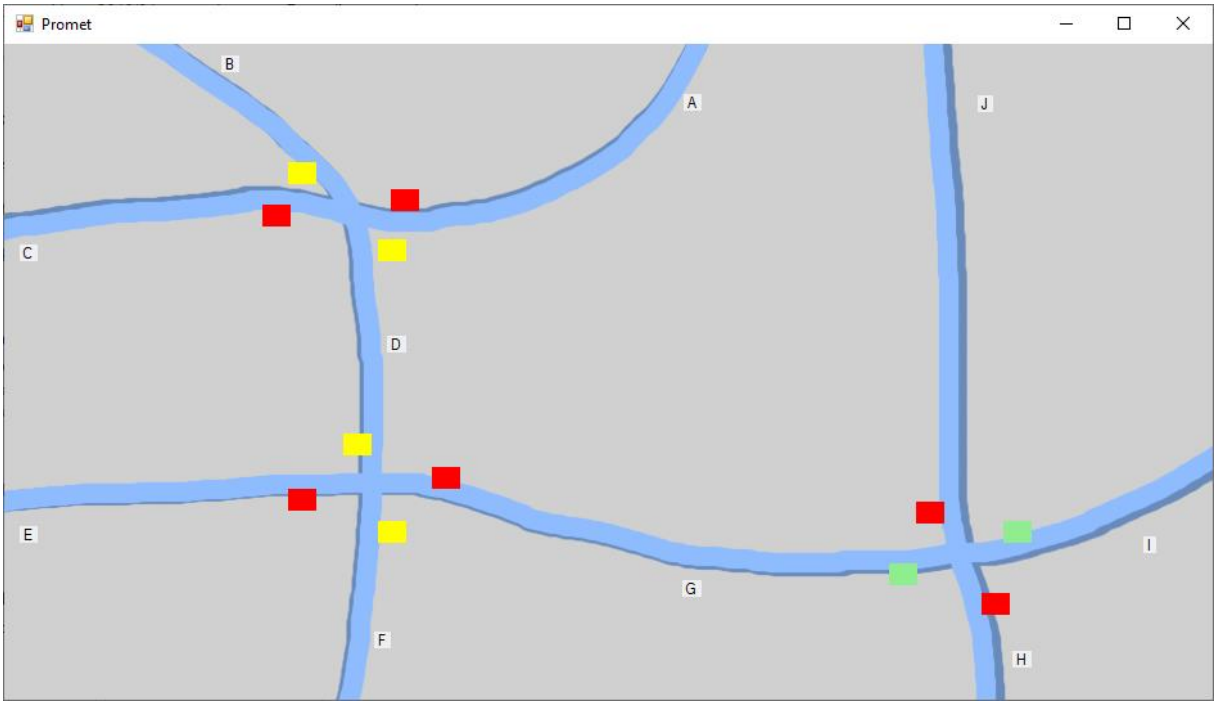
```
Incijaliziraj staraPoruka i novaPoruka, postavi vrijednosti prazni
string;
      Inicijaliziraj i pokreni tajmer;
      Na otkucaj tajmera radi
            Ako je(staraPoruka različito od novaPoruka)
                  Ispiši novaPoruka na ekran;
                  novaPoruka=staraPoruka;
      Sve dok tajmer radi
Na događaj Pritisak tipke spremi
      Spremi sadržaj ispisan na ekranu u datoteku;
Kraj procedure događaja
```

4.2. Prikaz rada simulacije

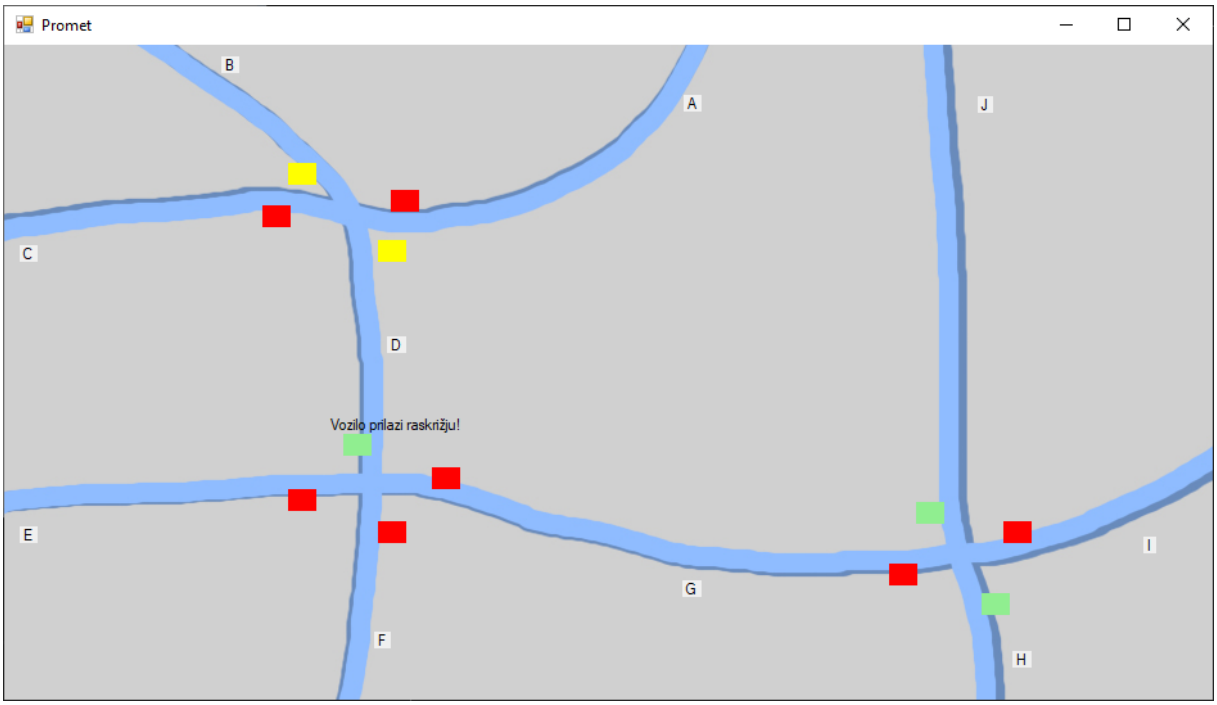
Prikaz rada logično je započeti formom koja prikazuje raskrižja koja rade u stanju bez prioritetnih događaja. Na slici broj 5 prikazana je forma s raskrižjima na kojoj se vidi kako su na sva tri raskrižja trenutno u funkciji, te na svakom svijetle po 2 zelena i 2 crvena semafora. Intervali raskrižja R1 i R2 se podudaraju, te se njihova svjetla izmjenjuju istovremeno, dok raskrižje R3 ima različiti interval. Na slici 6 vidi se promjena svjetala na raskrižjima R1 i R2 iz propuštanja jednog u propuštanje drugog smjera. Kod promjene se pale žuta svjetla. Na slici broj 7 vidljiv je rad simulacije u slučaju da postoji vozilo višeg prioriteta čije se kretanje simulira kroz sustav. Vidljivo je da se vozilo približava raskrižju R2 iz smjera pozicije D, odnosno u kodu simulacije R2 očitava prilazak sa sjevera. Kraj semafora na sjevernoj strani R2 dodana je tekstualna oznaka za lakše praćenje simulacije. Na ovaj način se pale i gase semafori na svim raskrižjima u generiranoj ruti za određeno vozilo.



Slika 5: Forma s raskrižjima u početku simulacije (isječak zaslona)



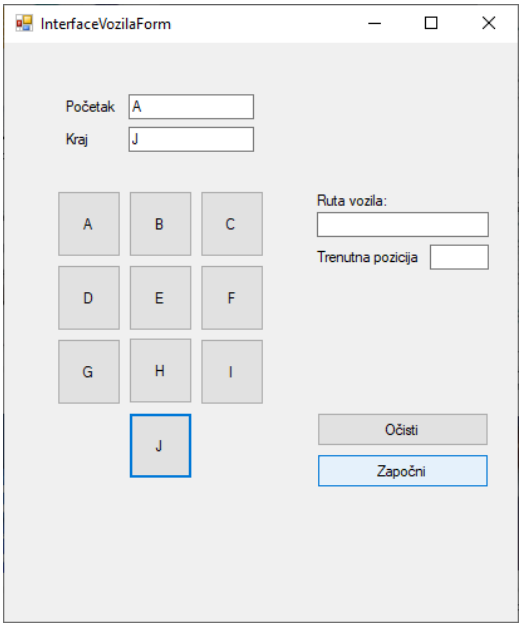
Slika 6: Forma s raskrižjima pri promjeni svjetala semafora (isječak zaslona)



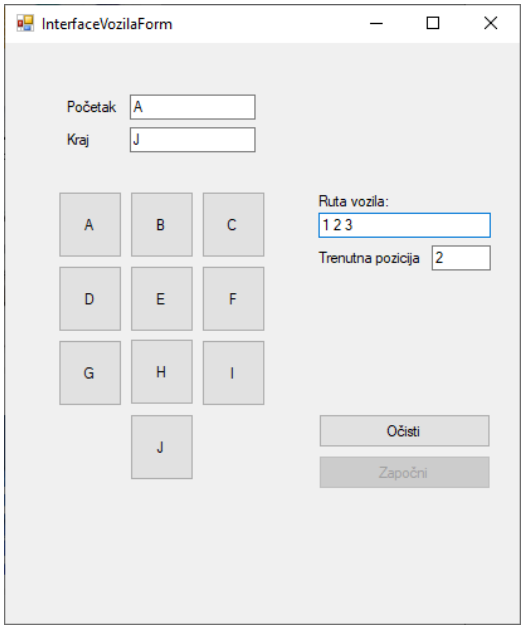
Slika 7: Forma s raskrižjima u prioritetnom modu (isječak zaslona)

Forma pomoću koje je moguće u simulaciji generirati kretanje vozila višeg prioriteta kroz raskrižja prikazana je na slikama 8 i 9. Na slici 8 vidimo formu na kojoj su klikom na gumbe A i J odabrani početak i kraj. Dodani početak i kraj mogu se lako obrisati klikom na gumb „Očisti“. Također je programski onemogućen unos iste pozicije za početak i kraj, odnosno ako odaberemo A kao početak, ponovnim pritiskom na A ne događa se ništa, a kraj ostaje nezabilježen. Na slici 9 vidimo formu nakon pritiskanja gumba „Započni“. Ako nisu uneseni potrebni parametri, ne započinje se izvođenje sljedećeg dijela programa, već korisnik dobiva poruku putem messageboxa. Ta poruka vidljiva je na slici 10. Ako su parametri uneseni program započinje generiranje rute, a gumb Započni se zaključava. Ovo osigurava da korisnik

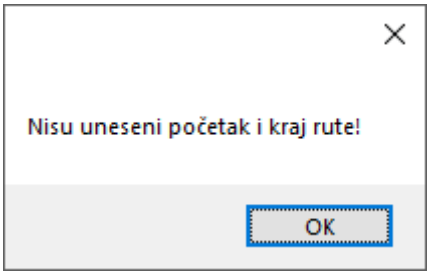
ne pokreće više simulacija prolazaka odjednom. Jednom kad vozilo stigne na završnu točku definiranu kao „kraj“, gumb će se ponovno otključati.



Slika 8: Forma sučelja vozila prije generiranja rute (isječak zaslona)

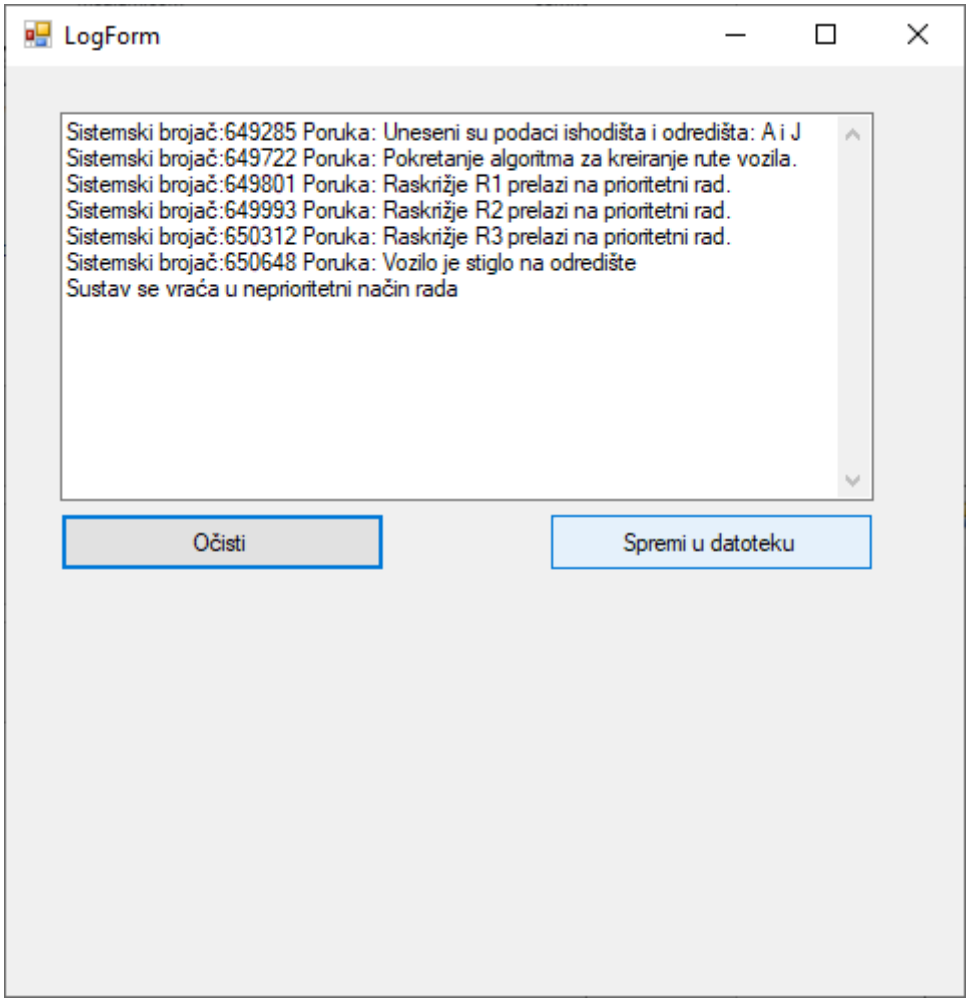


Slika 9: Forma sučelja vozila nakon generiranja rute (isječak zaslona)



Slika 10: Poruka koju korisnik dobiva ako nije unio parametre (isječak zaslona)

Posljednja forma u simulaciji prikazuje zabilješke o događajima u sustavu (kolokvijalno engl. log) na textbox-u u sredini forme. Te zabilješke generiraju se na nekoliko mjesta u programu, a njihovo slanje među klasama je prikazano u kodu u prethodnom poglavlju. Uz dio poruke koji govori o stanju simulacije u nekom trenutku dodan je i dio sa sistemskim brojače, koji pripaja određenoj poruci broj milisekundi kada je ona poslana nakon početka izvođenja simulacije. Pritiskom na gumb „Spremi u datoteku“ moguće je log spremiti u datoteku na računalu u .txt formatu. Pritiskom na „Očisti“ briše se dosadašnji log za textbox-a.



Slika 11: Forma s log-om događaja nakon prolaska kroz jednu rutu (isječak zaslona)

4.3. Nadogradnje sustava

S obzirom da simulacijom nisu obuhvaćene sve moguće funkcionalnosti ovakvog sustava, potrebno je napomenuti da one svakako postoje. Većini IoT sustava glavnu granicu određuju

budžet i kreativnost, a u povoljnim uvjetima prometnom sustavu s vozilom većeg prioriteta mogu se dodati sljedeći moduli.

- Nadogradnja izlaza

  Pod nadogradnjom izlaza postoje mogućnosti za dodavanje dodatnih svjetala semafora (strelice za skretanje), prikaz vremena trajanja svjetla, zvučnih signala za pješake i slično.
- Povezivanje na GPS

  Povezivanje na GPS bio bi osnovna razlika između onoga implementiranog simulacijom i stvarnog rješenja sustava. Vozilo koje se spaja na GPS putem njega automatski vidi svoju lokaciju, te koristi gotova rješenja za generiranje rute do odredišta. Osim toga GPS sustav u svakom trenutku zna lokaciju vozila, pa može precizno odrediti kada se semafori trebaju pali i gasiti kako bi se osiguralo optimalno trajanje prioritetnog rada svakog semafora. GPS sustav također može bilježiti podatke o prometnoj gužvi te čak pristupati podacima o vremenu ili radovima na cesti koji također mogu biti korisni za ovakav sustav.
- Više senzora

  Unutar simulacije nisu korišteni podaci o senzorima, jer ne uvelike utjecali na iskustvo korisnika, odnosno onoga koji promatra simulaciju. Međutim u stvarnom svijetu senzori svakom IoT sustavu otvaraju mogućnosti za ostvarenje većeg broja funkcionalnosti. Unutar raskrižja možemo koristiti senzore unutar sklopova koji očitavaju red čekanja na stranama raskrižja. Prema podacima s takvih senzora možemo mijenjati svjetla na semaforu i nastojati unutar sustava očuvati stanje sa što manje prometnih zastoja. Kao dio raskrižja moguće je implementirati i kamere za snimanje brzine ili pak senzore ispušnih plinova kako bi se vodila statistika na razini grada.

  Senzori postavljeni na raskrižja također mogu biti orijentirani i na pješake koji prilaze raskrižjima, pratiti frekvenciju pojavljivanja, brinuti o njihovoj sigurnosti i pratiti njihovo prosječno zadržavanje u sustavu jedno raskrižja. U slučaju uspješnog povezivanja prometnog sustava sa svakim pojedinim vozilom raskrižje bi moglo unaprijed upozoriti vozilo na ostale sudionike u prometu, što bi uvelike povećalo sigurnost u najtežim uvjetima kao što su snijeg i kiša.
- Povezivanje na bazu podataka

  Povezivanjem na bazu podataka od ovakvog jednostavnog sustava možemo vrlo brzo dostići Big Data sustav. Svaka vožnja vozila hitne pomoći generira podatke. Svako očitavanje senzora unutar sustava generira podatke. Nakon dovoljno vremena iz zapisa u bazi podataka moglo bi se otkrivati uzorke prisutne u sustavu te tako određivati npr. optimalna vremena trajanja svjetla na semaforu u određenom dijelu dana, brzinu micanja vozila ispred vozila hitne pomoći i slično.

- Povezivanje većeg broja raskrižja u sustav

  Uz implementaciju skalabilnih funkcionalnosti na početnom sustavu, njega bi bilo jednostavno proširiti na više raskrižja. Tako bi se postiglo veće područje kojim vozilo većeg prioriteta može prolazi neometano, a osim toga svakodnevno bi bilo generirano daleko više podataka za analizu i machine learning. S obzirom da gradovi često predstavljaju samo jednu prelaznu točku u prometnoj migraciji, te nisu sva prisutna vozila samo s tog gradskog područja, s većim brojem raskrižja bilo bi moguće pratiti utjecaj migracija u određenom dobu dana na opterećenje prometnica, a uz kvalitetnu analizu postoji mogućnost i prepoznavanja eventualnih ruta preusmjeravanja i sličnih mehanizama smanjenja opterećenja.

5. Zaključak

Interoperabilnost je jedna od osnovnih osobina interneta stvari koja ga čini sveprisutnim trendom koji rapidno raste u svim sferama života više od 15 godina unatrag. Internet stvari evaluirao je od svojih početaka prije 2010. godine kada se koristio relativno mali broj senzora u jednostavnim klijent-server sustavima, do današnjeg sveobuhvatnog spleta računalnih sustava koji međusobno razmjenjuju podatke, provode strojno učenje i koriste umjetnu inteligenciju.

Internet stvari krenuo je od RFID hardvera i relativno slabo-mobilnih senzora te se razvio do nano-tehnologija. Moderni IoT koristi self-learning i self-repairing, sustavi sve više postaju Things-To-Human orijentirani, a koristi za čovjeka javljaju se u sferama koje prije nije bilo moguće podržati IT-om.

Korištenje IoTa u područjima života kao što su briga za stare i nemoćne ili pak optimizacija prometa ili drugih javnih sustava, uvelike pridonose kvaliteti života. Kroz IoT moguće je ostvariti smanjenje troškova (primjer pružanja njege kod kuće), smanjenje vremena utrošenog na obavljanje kritičnih operacija (primjer prolaska vozila hitne pomoći sustavom) te generiranje novih znanja koristeći sustave strojnog učenja i umjetne inteligencije.

Zbog brzog razvoja IoT-a neki od glavnih problema koji se javljaju su potrebe za većim i bržim sustavima skladištenja podataka, potrebe za kvalitetnijim sustavima za obradu podataka i strojno učenje, te potreba za ostvarivanjem višeg stupnja sigurnosti i privatnosti za korisnika.

Popis literature

[1] Gubbi, J. Buyya R., Marusic Sl. (2013) Internet of Things (IoT): A vision, architectural elements, and future directions. *The University of Melbourne* (preuzeto 1.4.2020. s http://www.buyya.com/papers/Internet-of-Things-Vision-Future2013.pdf)

[2] Wamba S.F., The Electronic Product Code (EPC). *researchgate.net* (preuzeto 1.4.2.2020. s https://www.researchgate.net/figure/The-Electronic-Product-Code-EPC-example-of-a-96-bit-EPC-tag-Adapted-from-Leong-et_fig3_28149608)

[3] Hwang K., Chen M. (2017) Big-dana analytics for cloud, IoT and cognitive computing. *John Wiley & Sons* (preuzeto 1.4.2020. s https://books.google.hr/books?hl=hr&lr=&id=Kz1GDgAAQBAJ&oi=fnd&pg=PT10&dq=cloud+IoT&ots=b_GhL1nwj2&sig=nAWtkosrYffC_M6F9lW5XwaGKw4&redir_esc=y#v=onepage&q=cloud%20IoT&f=false)

[4] Lee I., Lee K. (2015) The Internet of Things (IoT): Applications, investments, and challenges for enterprises. *Business Horizons* (preuzeto 1.4.2020. s https://fardapaper.ir/mohavaha/uploads/2018/03/Fardapaper-The-Internet-of-Things-IoT-Applications-investments-and-challenges-for-enterprises.pdf)

[5] Chen S., Xu H., Liu D., Hu B.(2014) A vision of IoT: Applications, challenges, and opportunities with china perspective, *IEEE Internet Of Things* (preuzeto 1.4.2020. s https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6851114)

[6] List of GPS Satelites. *Wikipedia (dostupno 1.4.2020. na* https://en.wikipedia.org/wiki/List_of_GPS_satellites)

[7] Mishra A.(2017) Breadth First Search example (BFS) – How GPS navigation works. *Hackerearth.com* (dostupno 1.4.2020. na https://www.hackerearth.com/blog/developers/breadth-first-search-bfs-algorithm-example-gps-navigation/)

[8] Wagner P.(1996) Interoperability. *Brown University CRC Press* (preuzeto 1.8.2020. s https://dl.acm.org/doi/pdf/10.1145/234313.234424)

Popis slika

Slika 1:Razvoj IoT tehnologija od 2010. do danas (izvor: Gubbi et al 2013)..... 5
Slika 2: Elementi WSN sustava (izvor: http://www.ginsei-jp.com/wsn.html) 7
Slika 3: Faze različitih rješenja IoT arhitektura (izvor: https://www.researchgate.net/figure/The-4-Stage-IoT-Solutions-Architecture_fig12_329269702).....11
Slika 4: Prikaz sastavnica modela (vlastita izrada) 18
Slika 5: Forma s raskrižjima u početku simulacije (isječak zaslona).....28
Slika 6: Forma s raskrižjima pri promjeni svjetala semafora (isječak zaslona) 28
Slika 7: Forma s raskrižjima u prioritetnom modu (isječak zaslona).....29
Slika 8: Forma sučelja vozila prije generiranja rute (isječak zaslona).....30
Slika 9: Forma sučelja vozila nakon generiranja rute (isječak zaslona).....30
Slika 10: Poruka koju korisnik dobiva ako nije unio parametre (isječak zaslona) 30
Slika 11: Forma s log-om događaja nakon prolaska kroz jednu rutu (isječak zaslona).....31

Popis tablica

Tablica 1: Evolucija IoT-a (prema Sundmaeker et al (2010)) 4
Tablica 2: Veličine dijelova EPC taga (izvor: https://www.researchgate.net/figure/The-Electronic-Product-Code-EPC-example-of-a-96-bit-EPC-tag-Adapted-from-Leong-et_fig3_28149608) 6

Prilozi (1, 2, …)

[1] IoTpromet.sln datoteka C# projekta izrađenog u Visual Studiju 12.