MQTT i CoAP integracija Arduino uređaja sa thethigs.io cloud platformom

Studentski rad

Tanja Vukelić
Student drugog ciklusa studija
Elektotehnički fakultet, Univerzitet u Banjoj Luci
Banja Luka, BiH
tanjavukelic@gmail.com

Sažetak—**U radu je opisana implementacija MQTT (eng. *Message Queue Telemetry Transport*) i CoAP (eng. *Constrained Application Protocol*) klijenata sa Arduino Mega 2560 uređajem za direktno slanje senzorskih podataka na *thethings.io cloud* platformu. Dat je i pregled mogućih načina prikupljanja podataka sa thethings.io *cloud*-a za potrebe analize i integracije sa drugim sistemima. (Abstract)**

Ključne riječi-Internet of Things; MQTT; CoAP; Arduino; Cloud; (key words)

I. UVOD

Posljednjih godina došlo je do značajne ekspanzije neposrednih komunikacija između uređaja uz minimalno nadgledanje od strane čovjeka. Ovakav vid mreža, sačinjen od potencijalno velikog broja uređaja, danas popularno nazivamo Internet stvari – (eng. *Internet of Things – IoT*). U najširem smislu, IoT omogućava automatizovano prikupljanje, razmjenjivanje i obradu podataka te djelovanje na osnovu dobijenih informacija.

U IoT sistemima se od većine uređaja zahtijeva što veća efikasnost u smislu potrošnje energije i male cijene uređaja. Uzimajući u obzir ove kao i druge specifične komunikacione zahtjeve IoT sistema, pojavio se veliki broj novih protokola koji su optimizovani za IoT aplikacije, a u ovom radu biće opisani MQTT i CoAP.

Rad se sastoji od pet poglavlja. U drugom poglavlju ukratko su opisani IoT protokoli, MQTT i CoAP. Treće poglavlje je rezervisano za IoT platformu, dok je u četvrtom dijelu opisana implementacija klijenata za oba navedena protokola.

II. IOT PROTOKOLI

Aplikativni sloj nalazi se na vrhu IoT protokol steka i definiše metode za razmjenu aplikativnih podataka između krajnjh čvorova. Izbor protokola na aplikativnom sloju nije nimalo lak, jer uvijek promjenljive i zahtjevnije potrebe korisnika vode ka definisanju novih ciljeva, kao i razvoju savršenijih i usavršavanju postojećih IoT protokola.

A. MQTT

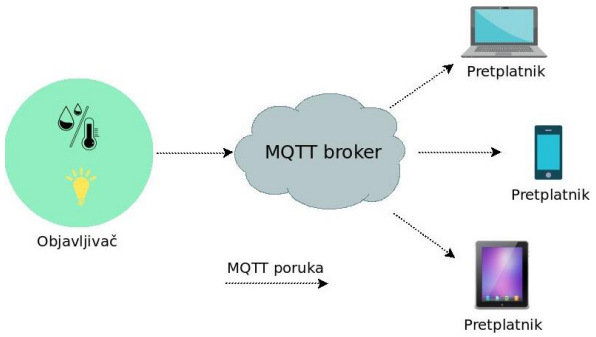
MQTT je protokol zasnovan na *Publish*/*Subscribe* (objava/pretplata) modelu razmjene podataka koji podrazumijeva postojanje brokera. Dizajniran je za uređaje sa ograničenim resursima kojima treba da omogući efikasan prenos podataka uz malu potrošnju [1].

Dva osnovna pojma, u okviru MQTT protokola [2], su „tema“ (eng. *Topic*) i „sadržaj“ (eng. *Payload*) poruke. Protokol funkcioniše po principu pretplate i objave: kada su različiti učesnici u komunikaciji zainteresovani za određenu temu, oni će se na istu pretplatiti ili objaviti određeni sadržaj. Broker je centralna komponenta mreže, odnosno poslužilac u komunikaciji, a klijenti su korisnici specifičnih usluga brokera. Sve poruke dolaze do brokera koji služi da ih pri pristizanju preusmjeri onim klijentima koji su na temu te poruke pretplaćeni. Na taj način je razdvojeno slanje poruka od njihove isporuke. Princip komunikacje MQTT protokola je prikazan na Sl. 1. MQTT kao protokol aplikativnog nivoa koristi TCP (eng. *Transmisson Control Protocol*) kao transportni protokol i tako nasljeđuje sve njegove dobre osobine kao što su garantovana isporuka kroz potvrde i retransmisije, kontrola brzine prenosa i dr. [3]. Pored toga, MQTT uvodi još tri tipa takozvanog „kvaliteta usluge“ (QoS, eng. *Quality of Service*):

- QoS-0: Poruka se šalje samo jednom i ne očekuje se potvrda o njenom prijemu (eng. *fire and forget*). U slučaju nestabilnosti u mreži moguće je da paket nikada ne stigne do brokera/pretplatnika, tj. sa QoS 0 MQTT nudi isti nivo garancija dostavljanja poruke kao i korišteni transportni protokol, TCP.
- QoS-1:Pošiljalac poruke očekuje potvrdu od primaoca za svaku poslanu poruku. Pošiljalac čuva poruku sve dok ne dobije potvrdu prijema za istu. Ukoliko potvrda prijema za neku poruku ne stigne u razumnom vremenskom okviru pošiljalac ponovo

šalje tu poruku. Zbog ovoga se može destiti da jedna poruka bude više puta dostavljena pa se stoga način prenosa poruka ovim QoS-om naziva *at least once*. Ovim QoS-om dodaje se još jedan nivo kontrole dostavljanja poruka pored TCP mehanizama.

- QoS-2:Ovo je najviši nivo usluge definisan u MQTT-u. Garantuje se da će poslana poruka samo jednom stići do primaoca (eng. *exactly once*). Za svaku poruku potrebno je razmijeniti najmanje četiri poruke između pošiljaoca i primaoca pa je ovaj metod prenosa, iako najsigurniji, ujedno i najsporiji.



Slika 1. *Princip komunikacije MQTT protokola*

B. CoAP

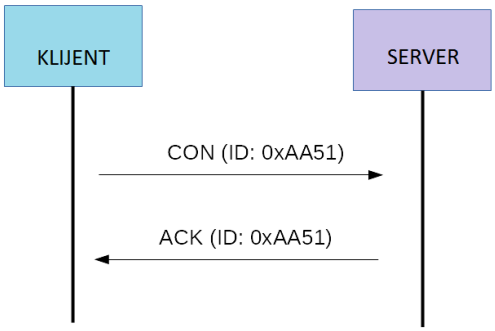
CoAP je protokol namijenjen za upotrebu u veoma jednostavnom hardveru (kao što su 8-bitni mikrokontroleri, senzori male snage i slični uređaji). On predstavlja realizaciju REST (eng. *Representational State Transfer*) arhitekture u formi pogodnoj za ograničene uređaje i mreže (mrežna ograničenja se odnose na brzinu prenosa, visok nivo gubitaka paketa, nedostatak naprednih servisa kao što je IP multicast, ograničenu veličinu paketa, ograničenu dostupnost uređaja u mreži itd.).

Popularni Internet protokoli, kao što su HTTP (eng. *Hypertext Transfer Protocol*) ili TLS (eng. *Transport Layer Security*) zbog svoje kompleksnosti nisu pogodni za rad sa uređajima koji imaju izražena hardverska ograničenja. Neke karakteristike CoAP-a su veoma slične HTTP-u, ali je CoAP posebno dizajniran za IoT. Kao transportni protokol za CoAP je izabran UDP (eng. *User Datagram Protocol*) koji je mnogo jednostavniji protokol od npr. TCP-a (manja zaglavlja, nema retransmisija i potvrda) pa je i na ovaj način napravljena ušteda u potrebnom propusnom opsegu [4].

CoAP, kao i HTTP, predstavlja klijent/server model komunikacije zasnovan na razmjeni zahtjeva i odgovora. On podržava četiri različita tipa poruka:

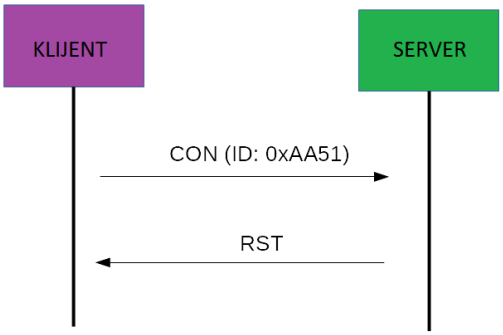
- Potvrda se zahtijeva (eng. *Confirmable, CON*)
- Potvrda se ne zahtijeva (eng. *Non-Confirmable, NON*)
- Potvrda (eng. *Acknowledgment, ACK*)
- Resetovanje (eng. *Reset, RST*)

Kada se zahtijeva pouzdan prenos informacije koristi se tip poruke *Confirmable* (CON). Koristeći ovaj tip poruke pošiljalac očekuje potvrdu od primaoca. Ukoliko u konfigurabilnom periodu potvrda ne stigne pošiljaocu poruka se šalje ponovo (maksimalan broj retransmisija je takođe konfigurabilan) dok druga strana ne pošalje poruku potvrde (ACK). ACK poruka sadrži isti ID kao i poruka koju potvrđuje (Sl. 2).



Slika 2. *Pouzdan prenos informacije*

Ako server ima internih problema u vezi sa prihvatanjem/obradom dolaznog zahtjeva, on može poslati poruku Reset (RST) umjesto poruke potvrde (ACK) i na taj način signalizirati pošiljaocu da je potrebno kontekst prenosa te poruke izbrisati i eventualno početi iz početka (Sl. 3).



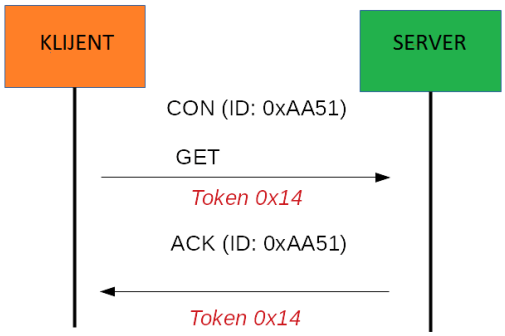
Slika 3. *Prikaz RST poruke*

U drugoj kategoriji su poruke koje ne zahtijevaju potvrdu od strane primaoca (NON). Ovakav tip poruka se može koristiti u slučajevima kada nije neophodno garantovati prenos informacije od pošiljaoca do primaoca (npr. senzorski podaci koji se često šalju). Ovakav način prenosa je brži, iako manje pouzdan, u odnosu na *Confirmable* (CON) način.

C. *Pojašnjenje modela zahtjev/odgovor kod CoAP protokola*

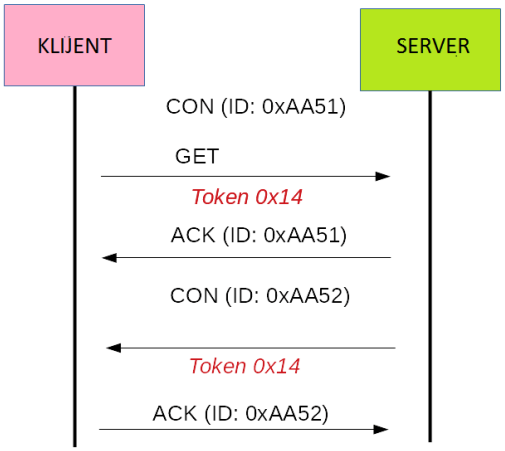
Zahtjev se šalje pomoću *Confirmable* (CON) ili *Non-Confirmable* (NON) poruke. Postoji nekoliko scenarija u zavisnosti od toga da li server može odmah da odgovori na zahtjev.

Ako server može odmah da odgovori na zahtjev klijenta i ako se zahtjev prenosi pomoću poruke *Confirmable* (CON), server šalje nazad klijentu poruku potvrde (ACK) koja u sebi sadrži i odgovor na zahtjev. ACK poruka ima uvijek isti ID kao i poruka na koju se odnosi (Sl. 4).



Slika 4. *Odgovor zajedno sa potvrdom*

Kao što se može primjetiti u CoAP poruci postoji i token. Token se razlikuje od ID-a poruke i koristi se da logički poveže zahtjev i odgovor (moraju imati isti token). Ako server ne može odmah odgovoriti na zahtjev koji dolazi od klijenta, on šalje poruku potvrde signalizirajući prihvatanje zahtjeva. Čim odgovor bude dostupan, server šalje klijentu novu potvrdivu (CON) poruku koja sadrži odgovor. Nakon toga, klijent šalje poruku potvrde (ACK) signalizirajući prijem odgovora (Sl. 5).



Slika 5. *Naknadni odgovor*

Ako se zahtjev koji dolazi od klijenta prenosi pomoću poruke koja ne zahtijeva potvrdu (NON), onda server također odgovara pomoću poruke koju nije potrebno potvrditi. U ovom slučaju ID poruke odgovora nije isti kao ID poruke zahtjeva, dok im je token isti.

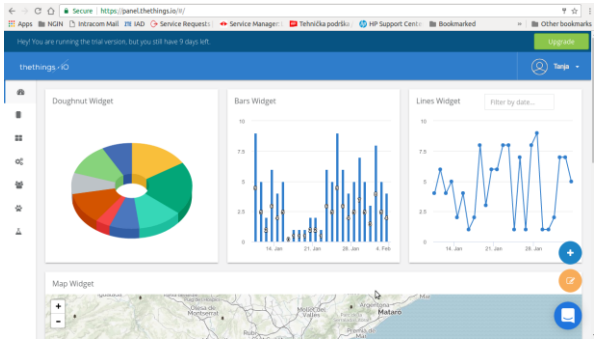
III. IoT PLATFORMA

A. *Cloud thethings.io*

Thethings.io je IoT platforma koja omogućava brzo i skalabilno povezivanje uređaja, njihovo praćenje i upravljanje u realnom vremenu te dobijanje raznih analitičkih izvještaja na osnovu prikupljenih podataka [5].

Thethings.io predstavlja presjek tri funkcionalnosti [5]:

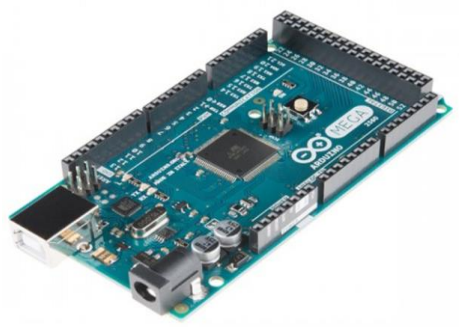
- IoT *Back-end* - rješenje za skladištenje podataka, ažuriranje *firmware*-a i upravljanje povezivanjem uređaja na veoma jednostavan način
- *Dashboards* - kontrolne table za IoT koje su potpuno fleksibilne (Sl. 6)
- *Big Data/Analitics* - alati za obradu prikupljenjih senzorskih podataka



Slika 6. *Primjer dashboard widget-a thethings.io platforme*

B. *Arduino platforma*

Arduino je platforma otvorenog koda (eng. *Open-source*) namijenjena za modelovanje i izgradnju elektronskih uređaja [6]. Arduino platforma obuhvata *hardware* u obliku razvojnih ploča s mikrokontrolerima te pripadajućih dodataka koji se nazivaju štitovi (eng. *Shields*). Pored *hardware*-a Arduino obezbjeđuje i programsko razvojno okruženje pod nazivom Arduino IDE te neophodne programske biblioteke za pisanje programa. U ovom radu korišten je Arduino Mega 2560 uređaj (Sl. 7) čije su osnovne karakteristike date u Tabeli I.



Slika 7. *Uređaj Arduino Mega 2560*

TABELA I . *KARAKTERISTIKE UREĐAJA ARDUINO MEGA 2560*

Mikrokontroler	Atmega 2560
Radni napon	5 V
Napon napajanja (preporučeni)	7-12 V
Maks. napon napajanja	20 V
Digitalni ulazno/izlazni pinovi	54 (14 PWM izlaza)
Analogni ulazi	16
DC struja po ulazno/izlaznim pinovina	40 mA
DC struja za 3,3V pin	50 mA
Flash memorija	256 kB
SRAM	8 kB
EEPROM	4 kB
Takt	16 MHz

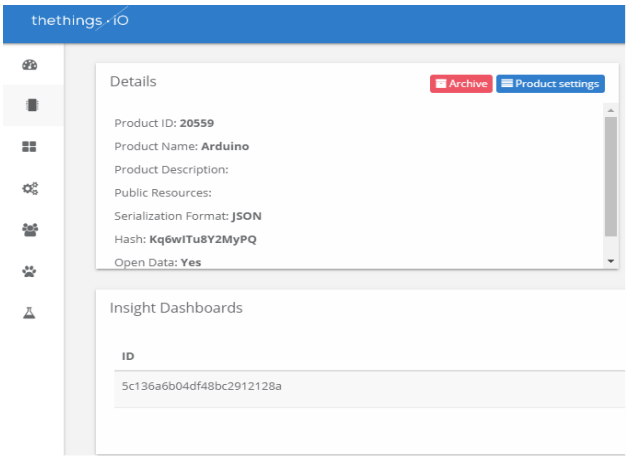
IV. IMPLEMENTACIJA

A. *Programiranje Arduino uređaja*

Za programiranje Arduino Mega 2560 uređaja potrebno je USB kablom (USB 2.0 tip A/B) ploču povezati sa računarom (u ovom slučaju korišten je Fujitsu Lifebook E557 sa OS WIN 10 Enterprise). Za povezivanje Arduina na Internet korišten je *Ethernet shild*. Za razvojno okruženje, pisanje i prevođenje koda te prebacivanje dobijeng binarnog koda na Arduino korišten je Arduino IDE verzija 1.8.5. Korišten je digitalni senzor za mjerenje temperature i vlažnosti, *Adeept DHT11*.

B. *Konfigurisanje na strani thethings.io cloud-a*

Pomoću *cloud* korisničkog web interfejsa (Sl.8) u sklopu sekcije *Things Manager* potrebno je definisati *Product* koji možemo shvatiti kao grupu IoT uređaja i svaki korišteni uređaj aktivirati. Aktiviran uređaj dobija svoje atribute kao što su: ime (*Name*), ID (*Thing Id*) i token (*Thing Token*).



Slika 8. thethings.io korisnički web interfejs

C. *Implementacija Arduino MQTT (v3.1.1) klijenta*

- Potrebno je programirati Arduino tako da djeluje kao MQTT klijent koji šalje mjerne podatke sa senzora DHT11 u JSON formatu na *cloud*
- JSON *payload* mora biti u zahtijevanom formatu definisanom od strane thethings.io API-a, kao:

  {"values":[{"key":"Key","value":"Value"}…]}

  npr. {"values":[{"key":"Temperatura","value":17},

  {"key":"Vlaznost","value":21}]}
- Prilagoditi parametar MQTT_MAX_PACKET_SIZE u okviru fajla PubsubClient.h koji se koristi za pisanje programskog koda (u zavisnosti od dužine JSON *payload*-a ako je potrebno) [7]
- Kao adresu MQTT brokera potrebno je postaviti: *mqtt.thethings.io* (korišten je standardni TCP port 1883).
- Tako pripremljene podatke potrebno je poslati (*Publish*) na *Topic*:

  v2/things/<Thing Token>

Korištena Arduino biblioteka PubSubClient podržava slanje (*Publish*) MQTT poruka sa QoS 0. Memorijska ograničenja su razlog zašto nisu implementirani QoS 1 i QoS 2, ali to ne predstavlja prevelik problem jer MQTT koristi TCP kao transportni protokol, a sam TCP je pouzdan.

D. *Implementacija Arduino CoAP klijenta*

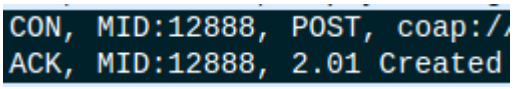
- Potrebno je programirati Arduino tako da djeluje kao CoAP klijent koji šalje mjerne podatke sa senzora DHT11 u JSON formatu na *cloud*
- JSON p*ayload* mora biti u zahtijevanom formatu definisanom od strane thethings.io API-a. Format JSON *payload*-a mora biti isti kao i u slučaju MQTT protokola koji je već opisan u prethodnoj sekciji
- Prilagoditi parametar BUF_MAX_SIZE u okviru fajla Coap.h koji se koristi za pisanje programskog koda (u zavisnosti od dužine JSON *payload*-a ako je potrebno) [8]
- Implementirati POST metod u okviru korištene biblioteke *CoAP-simple-library* [8], pošto ista metoda nije definisana u okviru navedene biblioteke (thethings.io zahtijeva da se podaci šalju upravo tom metodom)
- Kao adresu CoAP servera potrebno je postaviti: *coap.thethings.io* (korišten je standardni CoAP port 5683).
- Tako pripremljene podatke potrebno je poslati (POST metodom) na :

  coap://coap.thethings.io/v2/things/<Thing Token>

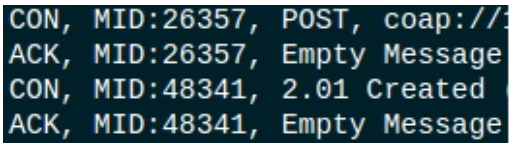
- Implementacija protokola na strani *cloud*-a dovodi do sljedećih mogućih odgovora na poslanu CoAP POST poruku:

1) *thethings.io* odgovara sa potvrdom prijema i šalje odgovor u okviru iste poruke (Sl. 9)
2) *thehings.io* prvo potvrđuje prijem zahtjeva, a naknadno šalje odgovor čiji je prijem potrebno potvrditi od starne klijenta (Sl. 10)

   Bilo je potrebno implementirati funkcionalnost pod 2) pošto ista nije bila u potpunosti definisana u korištenoj coap biblioteci.



Slika 9. *Primjer direktnog odgovora*



Slika 10. *Primjer naknadnog odgovora*

E. Prikupljanje i obrada senzorskihi podataka

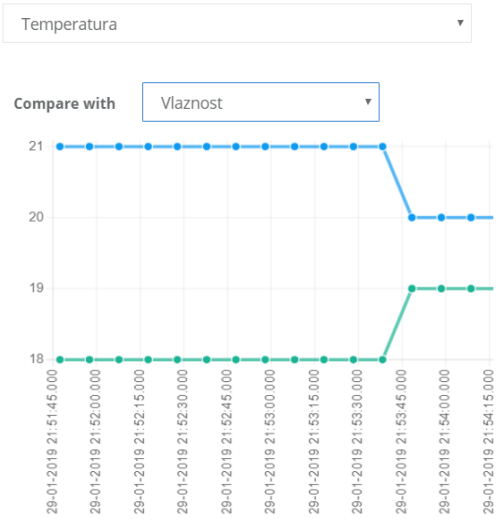
Thethings.io *cloud* podržava pisanje *cloud* funkcija u javascript programskom jeziku, prosljeđivanje poruka u realnom vremenu kao i pravljenje dashboard-a za krajnje korisnike.

Što se tiče prikupljanja podataka sa *cloud*-a i njihovog prosljeđivanja prema svim zainteresovanim stranama (npr. aktuatori, baze podataka, sistemi za naknadnu obradu...) postoje dva načina realizacije:

- u realnom vremenu
- istorijski

Slika 11. *Podaci kao odgovor na HTTP zahtjev*

U realnom vremenu podaci se prikupljaju tako što se zainteresovane strane MQTT protokolom pretplate na *Topic* koji referiše na token ID senzora koji im je interesantan ili se definisanim *cloud* funkcijama prosljeđuju zaintersovanim stranama. Slično, istorijski podaci se mogu pokupiti putem HTTP(S) zahtjeva (Sl. 11) ili periodično slati *cloud* funkcijama. Primjer vizualizacije isporijskih podataka prikazan je na Sl. 12.



Slika 12. *Vizualizacija istorijskih podataka na cloud-u*

V. ZAKLJUČAK

U ovom radu je opisana integracija uređaja Arduino Mega 2560 sa thethings.io *cloud* platformom, na bazi MQTT i CoAP protokola. Pokazano je da je integracija moguća i da thethings.io platforma nudi skup vrlo fleksibilnih alata i funkcionalnosti koji se mogu iskoristiti za širok spektar aplikacija. Takođe, thethings.io platformu lako je povezati sa drugim sistemima pa korisnik ove platforme nije ograničen samo na funkcionalnosti koje ona nudi. Thethings.io je jedna od rijetkih IoT *cloud* platformi koja podražava i CoAP i MQTT protokol. Važna karakteristika svakog IoT sistema je svakako sigurnost. Ovim radom to nije obuhvaćeno i korišten je nezaštićen vid komunikacije, u slučaju oba protokola. U budućem radu, bilo bi potrebno istražiti da li je moguće implementirati neki vid zaštite na korištenom Arduino uređaju imajući u vidu njegove *hardware*-ske karakteristike i ograničenja.

ZAHVALNICA

Posebnu zahvalnost bih izrazila prof. dr Gordani Gardašević pod čijim nadzorom i mentorstvom je izrađen ovaj rad.

LITERATURA

[1] http://mqtt.org/faq
[2] Banks, Andrew i Rahul Gupta. "MQTT Version 3.1. 1." OASIS Standard (2014).
[3] I. Bašićević, M. Popović, V. Kovačević, "Osnovi računarskih mreža 1", FTN, 2013.
[4] https://dzone.com/articles/coap-protocol-step-by-step-guide
[5] Thethings.io *Cloud*, Web site: https://thethings.io/
[6] https://www.arduino.cc/en/guide/introduction
[7] https://github.com/knolleary/pubsubclient
[8] https://github.com/hirotakaster/CoAP-simple-library

ABSTRACT

This paper presents an example of the implementation of MQTT (Message Queue Telemetry Transport) and CoAP (Constrained Application Protocol) clients with the Arduino Mega 2560 device for directly sending sensor data to thethings.io cloud platform. An overview of possible ways to collect thethings.io cloud data for analysis and integration with other systems is also provided.

MQTT AND COAP INTEGRATION OF ARDUINO DEVICES WITH THETHINGS.IO CLOUD PLATFORM

Tanja Vukelić