SmartAlarm - Sistem za monitoring zvuka

Aleksa Ilić, Andrija Cvetković

Sadržaj - Rad prikazuje razvoj ideje, projektovanje, izgradnju i programiranje uređaja koji vrši monitoring zvuka i na osnovu analize trajanja njegove jačine, donosi odluku o promeni stanja, nakon čega preko servisa *PushBullet* obaveštava korisnika putem notifikacije na mobilnom telefonu ili desktop računaru. *PushBullet* podržava sve vodeće operativne sisteme, te je ovaj pristup multiplatformski. Ukoliko korisnik ne želi da koristi ovaj servis (za koji je neophodna besplatna apikacija), njemu će umesto notifikacije biti prosleđen mail. Uređaj je povezan sa centralnim web serverom koji pruža korisniku, kako direktnu kontrolu nad njim, tako i mogućnost analize poruka koje on odašilje. Svaki uređaj poseduje jedinstveni serijski broj, pa jedan korisnik može da poseduje više uređaja i da ih nesmetano koristi.

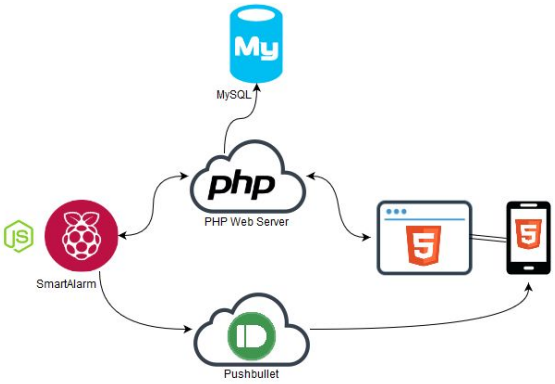
I. Uvod

Oblast mobilne telefonije i primenjene elektronike uopšte, proteklih godina je u naglom porastu i shodno tome sve je više aplikacija koje nam pružaju inteligentan vid automatizacije. Prativši te standarde došli smo do zaključka da nije bilo značajnih inovacija u oblasti opreme za bebe. SmartAlarm je pre svega konciprian kao uređaj koji putem interneta obaveštava roditelje o zvucima u dečjoj sobi i analizom izmerenih podataka utvrđuje da li je beba budna, što im umnogome olakšava život, nudeći pametno i efikasno rešenje za nadzor. Shvativši da SmartAlarm uz manje modifikacije može da poprimi širok spektar funkcija, odlučili smo da ga ovde predstavimo kao univerzalno rešenje za monitoring zvuka.

II. Princip rada

Za izradu SmartAlarm sistema upotrebljen je mikroračunar Raspberry Pi 1 Model B [1]. Na isti je povezana generička USB PnP Audio Kartica koja je nadomestila nepostojanje audio inputa na osnovnom modelu. Centralni web server, napisan u PHP-u i povezan sa MySQL bazom podataka nudi svojevrstan API alarmu koji mu pristupa dok korisniku servira multiplatformsku i interaktivnu web stranicu koja omogućava direktnu kontrolu nad alarmom. Svakom SmartAlarm-u se dodeljuje jedinstveni serijski broj koji ujedno služi i kao glavni identifikator serveru. Kada SmartAlarm zaključi da je nivo zvuka kontinuirano viši od prosleđenog parametra, prosleđuje tu informaciju serveru i servisu *PushBullet* [2], koji potom istu prosleđuje u vidu notifikacije ili putem elektronske pošte svim korisnicima.

Aleksa Ilić i Andrija Cvetković su učenici Gimnazije, Koste Stamenkovića 15, 16000 Leskovac, Srbija, E-mail: aleksa.d.ilic@gmail.com, c.andrija@yahoo.com



Slika 1. Šema rada

III. Server

Kod, napisan u PHP-u, usko je povezan sa MySQL bazom podataka i on korisnicima servrira administratorski panel dok uređaju omogućava pristup bazi. Sastoji se od nekoliko skripti:

- **index.php -** Početna stranica koja pruža osnovne informacije o dejstvovanju servera
- **register.php -** Stranica za registraciju korisnika
- **login.php -** Stranica za logovanje korisnika
- **dashboard.php - A**dministratorski panel
- **board_control.php** - Skripta koja upravlja uređajem i vraća JSON kao odgovor na upit
- **user_control.php** - Skripta koja omogućava dodavanje i izmenu uređaja
- **errors.inc.php** - Skripta koja omogućava upravljanje grešaka i sastavni je deo svake skripte
- **connect.inc.php** - Most između servera i baze koji je sastavni deo svake skripte

Pojedine skripte i interfejsi su detaljno opisane u narednim podsekcijama.

A. Baza podataka

MySQL baza podataka sastoji se iz 3 tabele: “users”, “boards” i “logs”.

Tabela “users” sadrži osnovne podatke o korisniku koji pristupa serveru, a popunjava se registracionom formom.. Takođe sadrži polje “serials” koje korisnik naknadno, kroz administratorski panel, popunjava serijskim brojevima svojih SmartAlarm-a.

Tabela registrovanih SmartAlarm-a (“boards”) je najznačajnija jer predstavlja most između korisničkog administratorskog panela i samog uređaja. Njena struktura

serial (TEXT)	status (TINYINT)	name (TEXT)	location (TEXT)	sensitivity (INT)
D8BS-L3V1-0915	1	Aleksin alarm	Aleksina soba	17

id (INT)	board (TEXT)	message (TEXT)	timestamp (TIMESTAMP)
2	D8BS-L3V1-0915	SmartAlarm skenira	2015-10-04 16:21:55
3	D8BS-L3V1-0915	Buffer iznosi: [24,2992,1999,59,79]	2015-10-04 16:22:00

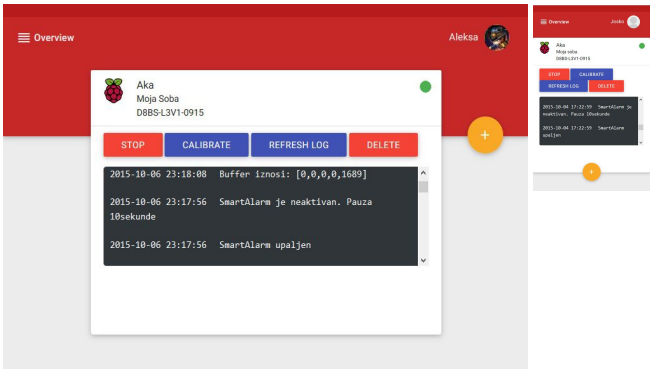
Tabela 1

je detaljno izložena u Tabeli 1. Primarni ključ je upravo serijiski broj jer je jedinstven za svaki alarm.

Tabela "logs" omogućava SmartAlarm-u da u realnom vremenu obavesti korisnika o njegovim deleatnostima. Povezana je preko stranog ključa "board" sa tabelom "boards", dok "id" predstavlja primarni ključ. Njena struktura takođe je detaljno izložena u Tabeli 1.

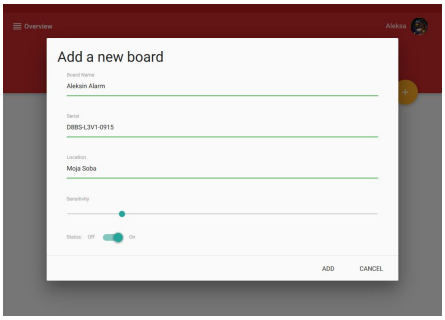
.

B. Dashboard



Slika 2. Dashboard na desktopu i mobilnom telefonu

Administratorski panel, prikazan na Slici 2, u realnom vremenu, pruža korisniku direktan uvid u stanje uređaja, a pritom mu omogućava da njime i upravlja. Izrađen u stilu Google-ovog "Material" dizajna [3], pruža fluidan interfejs svim platformama, a pomoću servisa *PhoneGap* [4] izrađene su i aplikacije za Android, iOS, WindowsPhone i BlackBerry. Pritiskom na dugme plus, korisnik biva upoznat sa formom za upis novog uređaja, koja sadrži: polja za unos imena, serijskog broja i lokacije uređaja, slajder za kontrolu osetljivosti i prekidač za određivanje statusa uređaja. Nakon dodavanja uređaja, korisniku se pruža mogućnost promene statusa, kalibrisanja osetljivosti i prikaza poruka koje uređaj odašilje.



Slika 3. Dodavanje uređaja

C. Server API

Na Listingu 1 data je skraćena verzija skripte *board_control.php*

```php
<?php
require('includes/connect.inc.php');
header('Content-Type: application/json');
if( !check($_GET['serial']) || !check($_GET['action']) )
        terminate(ERR::SERVER_DATA);
$query_str = "SELECT * FROM `boards` WHERE `serial`='$serial'";
$query = $link->query($query_str) or terminate(ERR::QUERY_CODE);
$board = getObj($query,ERR::BOARD_SERIAL);
switch($_GET['action']){
        case 'getData':
                (...)
        case 'isActive':
                (...)
        case 'addLog':
                (...)
        case 'setStatus':
                (...)
        case 'setCalibration':
                (...)
        case 'getLog':
                (...)
        default:
                terminate(ERR::SERVER_DATA);
}
exit(json_encode($data));
?>
```

Listing 1. *board_control.php*

API vraća JSON objekat, u zavisnosti od prosleđene komande. U tabeli 2 prikazani su neki od mogućih zahteva. API je programski nezavisan u smislu da nije pisan za određenu jezičku platformu, te se vraćenim podacima može manipulisati u zavisnosti od zahteva programera, čime se povećava modularnost sistema.

IV. SmartAlarm Klijent

A. Hardver

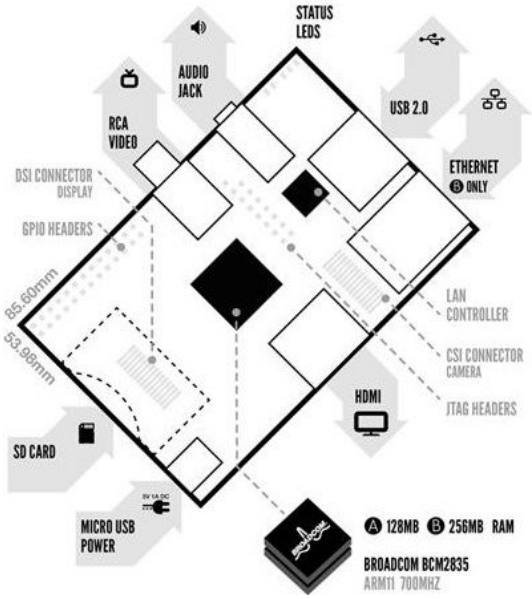
Za razvojnu platformu odabran je Raspberry Pi Model B [1] zbog svojih malih dimenzija, cene i modularnosti. Procesor je baziran na ARM arhitekturi i klokovan je na 700MHz. Internet je neophodan za funkcionisanje

HTTP request	JSON
?action=getData	`{"status":"0","calibration":"0","name":"Aka","location":"Moja Soba", "sensitivity":"17", "users": [{"email":"aleksa.d.ilic@gmail.com", "name":"Aleksa", "surname":"Ilic"}]}`
?action=addLog	`{"status":false,"errors":[{"code":"ERR_SERVER_NODATA","message":"Nedovoljno podataka prosledjeno"}]}`
?action=addLog&value=newLog	`{"status":true,"errors":[]}`
?action=setStatus&value=1	`{"status":true,"errors":[]}`
?action=getLog	`"status":true,"logs":[{"message":"Buffer iznosi: [123,1245,612,215,12]","timestamp":"2015-10-07 12:32:50"}, ... }`

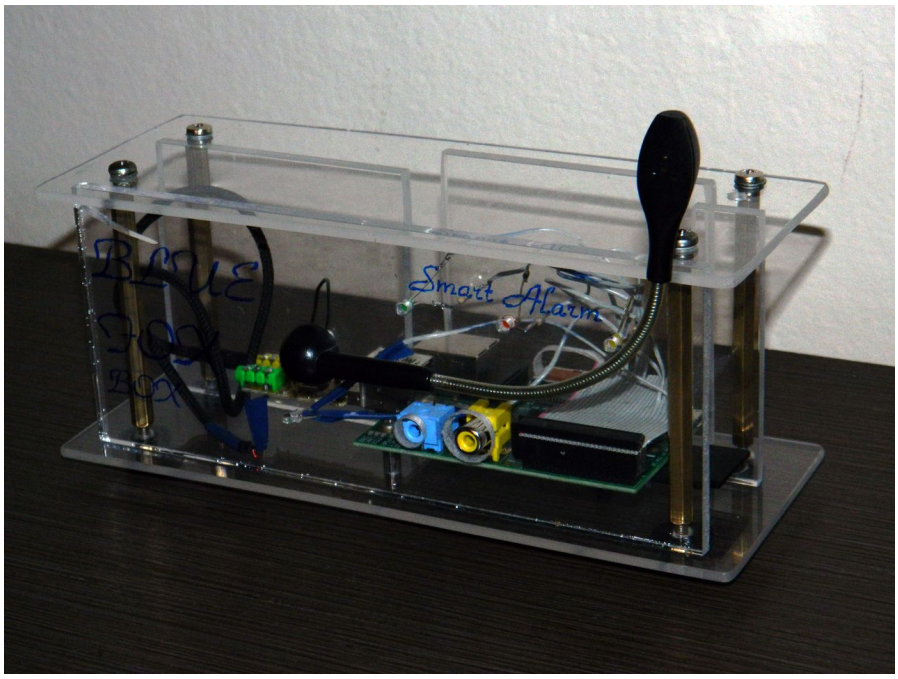
Tabela 2

SmartAlarm-a, te postojanje integrisane mrežne kartice omogućava lako povezivanje na mrežu. Na SD karticu, koja čini osnovnu memoriju računara, instaliran je Raspbian OS, prilagođena verzija Linux distribucije Debian, koja sadrži sve neophodne drajvere i pakete. Na GPIO pinove zakačene su LED diode koje signaliziraju rad alarma. Crvena svetli kada je alarm neaktivan, zelena kada je aktivan, a žuta treperi dok god uređaj komunicira sa spoljnim serverom. Nedostatak ove platforme je nepostojanje audio inputa što smo nadomestili ugradnjom generičke USB PnP Audio Kartice, čija će konfiguracija biti objašnjena u narednom poglavlju, a na nju je priključen kapacitivni analogni mikrofon.

Kućište prototipa izrađeno je od pleksiglasa, a potom učvršćeno pomoću spejsera i u potpunosti je ručno napravljeno. Napravljeni su otvori za eventualne nadogradnje (audio, video, HDMI). Mikrofon je postavljen tako da se može okretati za 360° u zavisnosti od potrebe. Kućište je potpuno modularno tako da se veoma lako može skinuti jedna od strana i umetnuti druga po potrebi.



Slika 4. Raspberry Pi - blok šema i priključci



Slika 5. Izgled prototipa

B. Softver

Za komunikaciju sa mikrofonom koji je povezan sa generičkom USB PnP Audio Karticom korišćeni su ALSA drajveri za zvuk koji su integrisani u Raspbian OS. Promenom indeksa podrazumevane zvučne kartice u konfiguracionam fajlu ALSA drajvera [5], omogućen je pristup mikrofonu. Skripta za ekstrakciju trenutne jačine zvuka napisana je u pythonu i koristi ALSA API [6] (Listing 2).

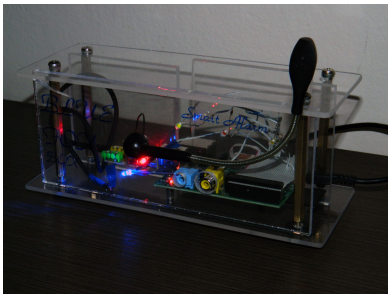
```
#!/usr/bin/python
import alsaaudio, time, audioop

inp=alsaaudio.PCM(alsaaudio.PCM_CAPTURE,
    alsaaudio.PCM_NONBLOCK)
inp.setchannels(1) #Mono
inp.setrate(8000) #8000Hz
inp.setformat(alsaaudio.PCM_FORMAT_S16_LE) #16bit
inp.setperiodsize(160)

while True:
        l,data=inp.read()
        if l:
                print audioop.max(data, 2)
                break
```

Listing 2. *getdB.py*

Pri svakom startu pokreće se bash skripta *startup.sh* koja otvara GPIO pinove, apdejtuje kod preko git-a i pokreće glavnu skriptu.



Slika 6. SmartAlarm aktivan

Skripta *main.js* napisana je u javascriptu i izvšava se preko node.js-a, te je bilo neophodno kompajlitari izvorni kod te platforme za ARM procesore. Node.js je platforma izrađena na JavaScript izvršnom okruženju Google Chrome-a za jednostavnu izradu brzih, skalabilnih mrežnih aplikacija. Node.js koristi vođen-događajima, neblokirajući U/I model koji ga čini laganim i efikasnim, savršenim za aplikacije koje rade u realnom vremenu i iziskuju veće količine prenosa podataka i koje se izvršavaju preko distribuiranih uređaja. Ova skripta čini okosnicu sistema i izrađena je po uzoru na razvojnu platformu *Wiring* [7] u smislu da se sastoji iz 2 glavne funkcije:

1. *setup()* - funkcija koja se pokreće jednom, samo na startu i koristi se za definisanje pojedinih početnih vrednosti.
2. *loop()* - funkcija koja se poziva u više navrata tj. dok god je klijent uključen.

Kod skripte može se grubo podeliti na nekoliko fragmenta:

1. Integracija spoljnih modula i deklarisanje promenljivih nužnih za rad.
2. Implementacija prototipnih funkcija radi lakšeg manipulisanja nizovima
3. Implementacija funkcija *setup* i *loop* koje su prethodno opisane

B.1. Integracija modula

Moduli, neophodni za funkcionisanje skripte, mogu se lako preuzeti preko *npm*-a koji predstavlja menadžer modula za node.js, a od verzije 0.6.3 je i u sastavu node softverskog paketa.

Modul *python-shell* omogućava pozivanje python skripti unutar node.js skripte. U našem slučaju, omogućava pozivanje skripte *getdB.py* koju smo opisali prethodno.

Modul *pi-gpio* omogućava kontrolu dioda povezanio na GPIO pinove uređaja.

Modul *urllib* omogućava asinhrono kreiranje HTTP zahteva tj. predstavlja vid komunikacije sa glavnim serverom.

Modul *pushbullet* pruža metode za slanje notifikacija svim uređajima koji su registrovani na mejl korisnika.

Za korišćenje servisa *PushBullet* neophodna je prethodna registracija radi dobijanja jedinstvenog API ključa, preko kojeg je moguće koristiti servis kroz kod.

B.2. Implementacija prototipnih funkcija

Funkcija *shiftpush* umeće novi element dok stari izbacuje.

Funkcija *check* proverava da li svi elementi buffer-a prelaze zadatu vrednost.

Funkcija *max* vraća najveći element niza.

Funkcija *isFull* proverava da li je niz popunjen ne nula vrednostima

B.3. Funkcije setup i loop

Funkcija *setup* otvara GPIO pinove za output i pali crvenu diodu. Nakon toga transmituje poruku serveru da je SmartAlarm aktiviran i pokrece funkciju *loop.*

Pored podešavanja osetljivosti uređaja parametrom *sensitivity*, promenom parametra *sleep* moguće je podesiti i njegovu inertnost što će sprečiti da uređaj promptno reaguje na šum ili slučajni zvuk kratkog trajanja. Kod

donosi odluku na bazi detektovanja 5 uzastopnih maksimalnih vrednosti iznad praga osetljivosti, u zavisnosti od 5 prethodno uzastopno uzorkovanih vrednosti pojedinačnog trajanja definisanog parametrom *sleep*. Ovo znači da minimanlno vreme neophodno za promenu stanja uređaja iznosi 25 *sleep* intervala.

Na listingu 3 data je skraćena glavna funkcija. Sa listinga je izbačeno definisanje promenljivih, pozivi serveru, kontrola GPIO pinova i paljenje dioda, logovanje koje se vraća serveru itd. Kompletan kod nalazi se na *github* stranici projekta [8]. Sledi kratka analiza iznetog koda:

Linija 2 proverava svojstvo *status* objekta *data* koji je definisan pozivom na serverski API koji vraća JSON.
Linija 3 vrši poziv python skripte za analizu jačine zvuka (Listing 1)

Kod u linijama 8-10 proverava poopunjenost *prebuffer-a* i ukoliko jeste unosi najveći element dok *prebuffer* resetuje.

Linija 13 proverava da li je svaki element niza *buffer* veći od broja definisanog parametrom *sensitivity* u administratorskom panelu.

Petlja na linijama 14-16 prosleđuje zahtev *PushBullet-u* da započne emitovanje notifikacija svim korisnicima koji poseduju SmartAlarm.

Linija 17 poziva funkciju koja prosleđuje serveru zahtev za promenu statusa i prosleđuje funkciju koja će biti pozvana nakon odogovora servera.

Linija 23 Izračunava vreme utrošeno za izvršavanje prethodnih komandi i na osnovu njega i parametra *sleep,* određenog u fragmentu 2, te odlaže poziv glavne funkcije za taj interval. Ovaj postupak omogućava konstantno uzorkovanje nezavisno od brzine hardvera.

```
1.  (...)
2.  if(data.status == 1){
3.      var getdB=new py('getdB.py',{mode:'text',scriptPath:"/home/pi/alarm/"});
4.      var startDate = new Date();
5.      getdB.on('message',function(dB){
6.          dB=parseInt(dB);
7.          prebuffer.shiftpush(dB);
8.          if(prebuffer.isFull()){
9.              buffer.shiftpush(prebuffer.max());
10.             prebuffer= new Array (0,0,0,0,0);
11.             trigger=true;
12.         }
13.         if(buffer.check(THRESHOLD)){
14.             for(var i=0;i<data.users.length;i++){
15.                 PUSHER.note(data.users[i].email,'SmartAlarm','Beba je budna.');
16.             }
17.             changeStatus(function(){
18.                 buffer = new Array(0,0,0,0,0);
19.                 loop();
20.             });
21.         }else{
22.             var stopDate=new Date();
23.             var diff=SLEEP-(stopDate-startDate);
24.             diff>0 ? setTimeout(loop,diff) : setTimeout(loop,0);
25.         }
26.     });
27. }else{
28.     setTimeout(loop,10000);
29. }
```

Listing 3. *main.js*

V. ZAKLJUČAK

Projekat je u potpunosti realizovan, alarm je ispunio sve prethodno zadate zahteve.

Ovim radom smo tek zagrebali površinu i ovo je samo dokaz valjanosti koncepta. Kao što smo već napomenuli, zahvaljujući svojoj modularnoj prirodi, SmartAlarm, uz manje modifijackije, može poprimiti širok spektar funkcija. Dodatnom analizom baferovanih vrednosti može se utvrditi da li nivo zvuka raste/opada, ili kolika je njegova srednja vrednost, što otvara vrata za korišenje uređaja za složenije potrebe kao što su merenje Doplerovog efekta, merenje dolazećeg i odlazećeg zvuka, u ekološke svrhe u vidu detektora nivoa buke na ulicama, kao i u drugim prirodno i društveno korisnim situacijama.

ZAHVALNICA

Autori se zahvaljuju profesoru informatike, Andriji Tričkoviću, koji ih je uputio na zajednički rad i podržavao tokom razvojnog procesa, kao i dobrom prijatelju Bogdanu Baljozoviću za pruženu moralnu podršku, onda kada im je ona bila neophodna.

REFERENCE

[1] Specifikacija platforme Raspberry Pi 1 Model B. Dostupan na: http://www.raspberry-projects.com/pi/pi-hardware/raspberry-pi-model-b/hardware-general-specifications

[2] Dokumentacija servisa Pushbullet. Dostupna na: https://docs.pushbullet.com/

[3] Specifikacija "Material" dizajna. Dostupna na: https://www.google.com/design/spec/material-design/introduction.html

[4] PhoneGap servis. Dostupan na: http://phonegap.com/

[5] Uputstvo za konfiguraciju ALSA drajvera. Dostupno na: http://plugable.com/2014/11/06/how-to-switch-to-usb-audio-on-raspberry-pi

[6] Dokumentacija ALSA API-a za jezik python. Dostupna na: http://larsimmisch.github.io/pyalsaaudio/libalsaaudio.html

[7] Opis razvojne platforme Wiring. Dostupan na: https://en.wikipedia.org/wiki/Wiring_%28development_platform%29

[8] Github stranica projekta. Dostupno na: https://github.com/aleksailic/smart-alarm-client https://github.com/aleksailic/smart-alarm-server