UNIVERZA NA PRIMORSKEM
FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN INFORMACIJSKE TEHNOLOGIJE

Tehnična dokumentacija

Pametna Hiša

Boštjan Mihelčič De Tommasi, Klemen Krnel

Doc v1.0

Koper 2017

Kazalo

1. Uvod

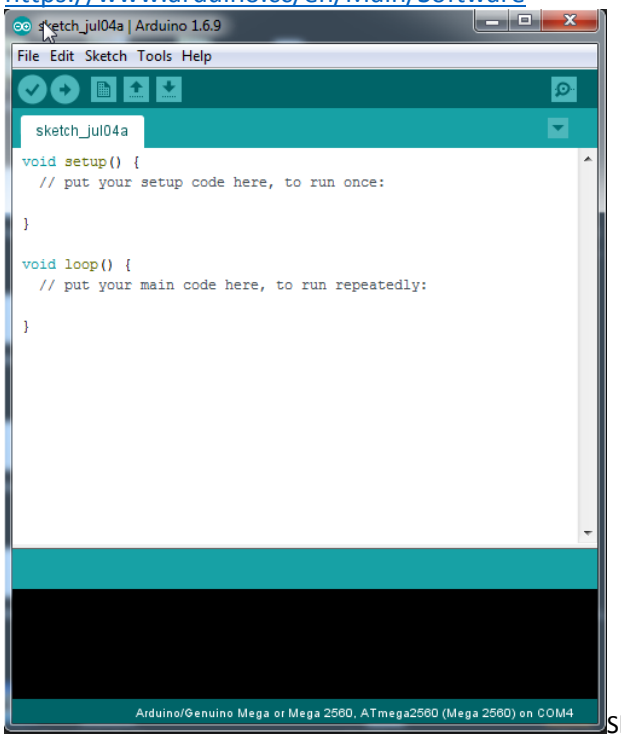
Tehnična dokumentacija vsebuje opis osnovnega sistema pametne hiše in potrebne informacije za izgradnjo in razširitev sistema. Za razumevanje dokumetacije je potrebno neko predzanje iz programskih jezikov ter elektronike. Dokumetacija obsega le razlago ki je specifična za sistem pametne hiše, ostale tehnične podrobnosti ki so prosto dostopne na internetu presegajo okvir dane dokumentacija.

1.1 Opis problema

Razvoj tehnologije nam je precej olajšal življenje. Npr. nam ni treba več prati posode saj nam jo stroj opere, ni nam treba več prati perila saj tudi to opravi stroj namesto nas. Ideja pametne hiše je nastala zato, da bi to lagodje še nadgradili. Ne bo nam potrebno več vstat za it dvignit ali spustiti rolete, klima se bo samodejno prižgala. Tudi ob vstopu v stanovanje bo luč že prižgana, če bo temno v prostoru. Vse bo povezano na skupno napravo tako da za vse naprave bomo rabili samo en upravljalec.

2. Razvojno okolje

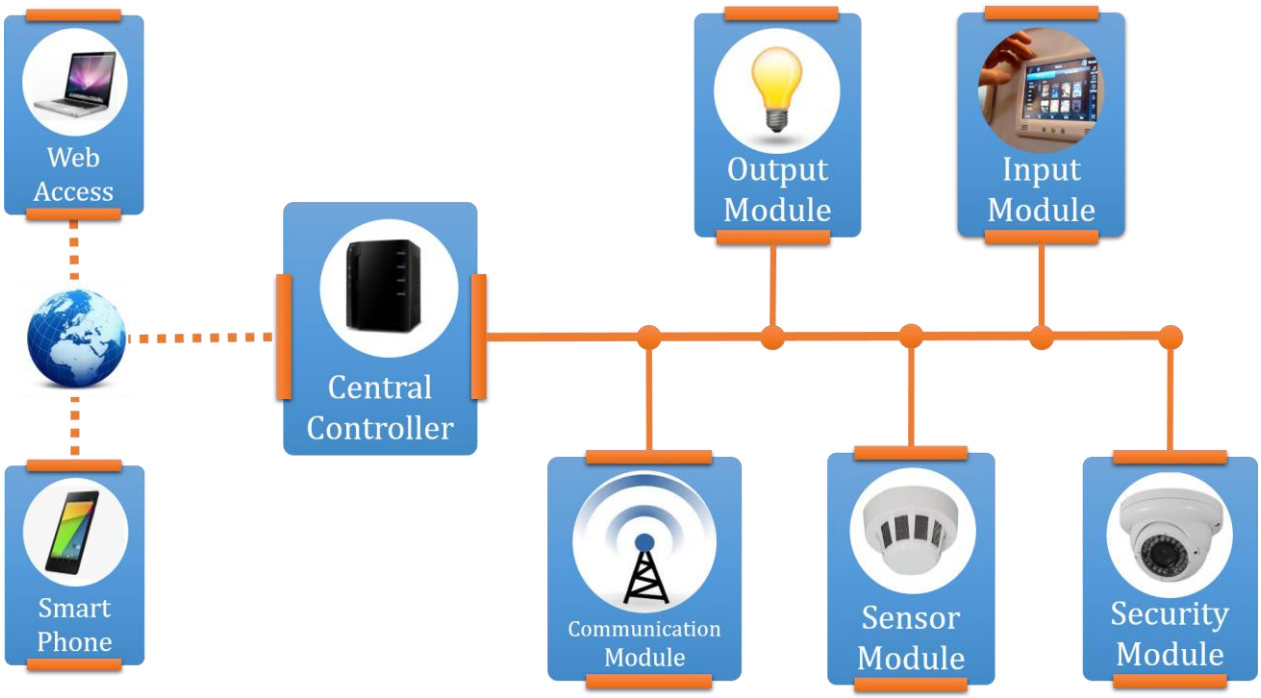
Pametna hiša je razdeljena na Rasberry Pi platformo ter Arduino razvojno ploščo.
Program za Raspberry Pi je napisan v Node.js v7.3.0, ki teče na Raspbian OS.
https://nodejs.org/en/
Za Arduino (slika 1) je uporabljeno prosto dostopno razvojno okolje Arduino 1.6.5 za različne operacijske sisteme na naslovu:
https://www.arduino.cc/en/Main/Software



Slika 1

3. Model

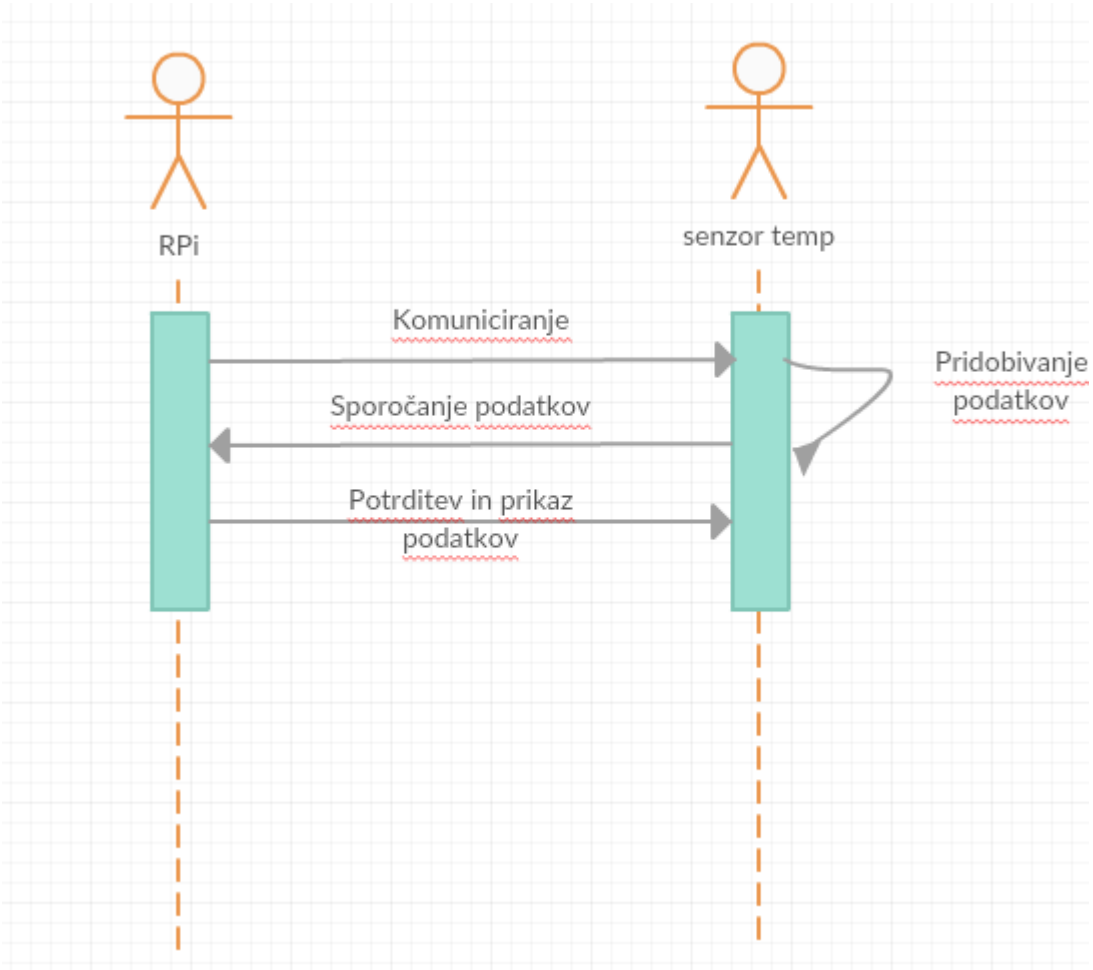
Za izvedbo projekta je bilo treba prej »skicirat« kako sploh poteka komunikacija med moduli (arduino) in raspeberry Pi. To je bilo začrtano z diagramom zaporedij (sequence diagram).
Spodnja slika (simbolična slika; slika 1) prikazuje kako naprave med seboj komunicirajo. Central Controller je v našem primeru RPi. Vsi moduli in senzorji sporočajo stanje RPi, ta pa vzpostavi ip naslov do katerega dostopaš preko računalnika ali pa pametnega telefona (oziromo vsako napravo ki ima brskalnik).



Slika 2

3.1 Diagram senzorja temperature

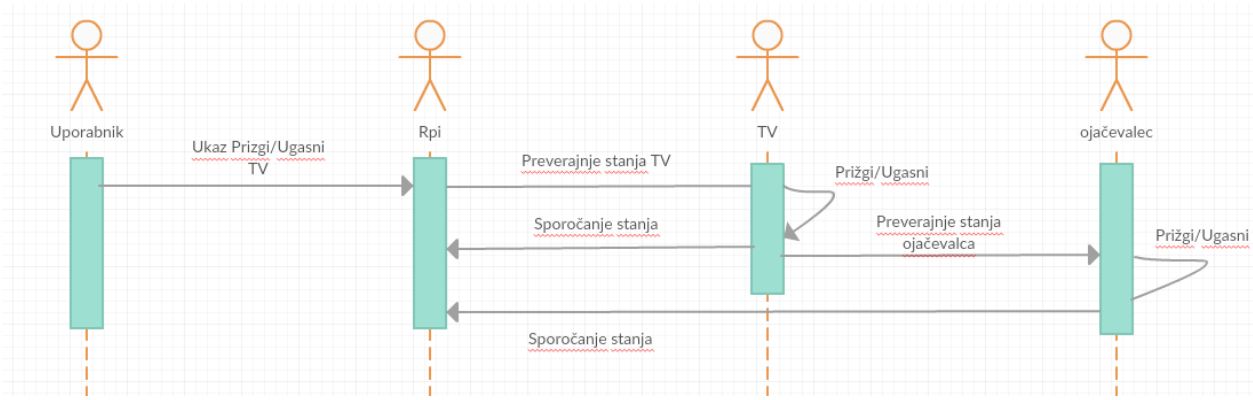
Raspberry Pi komunicira s senzorjem oregon, ta pridobi temperaturo jo prikaže in sporoči nazaj RPI stanje (slika 3).



Slika 3 – diagram zaporedja ki prikazuje komunikacijo med RPi in senzorjem temperature Oregon.

3.2 Diagram AV TV modula

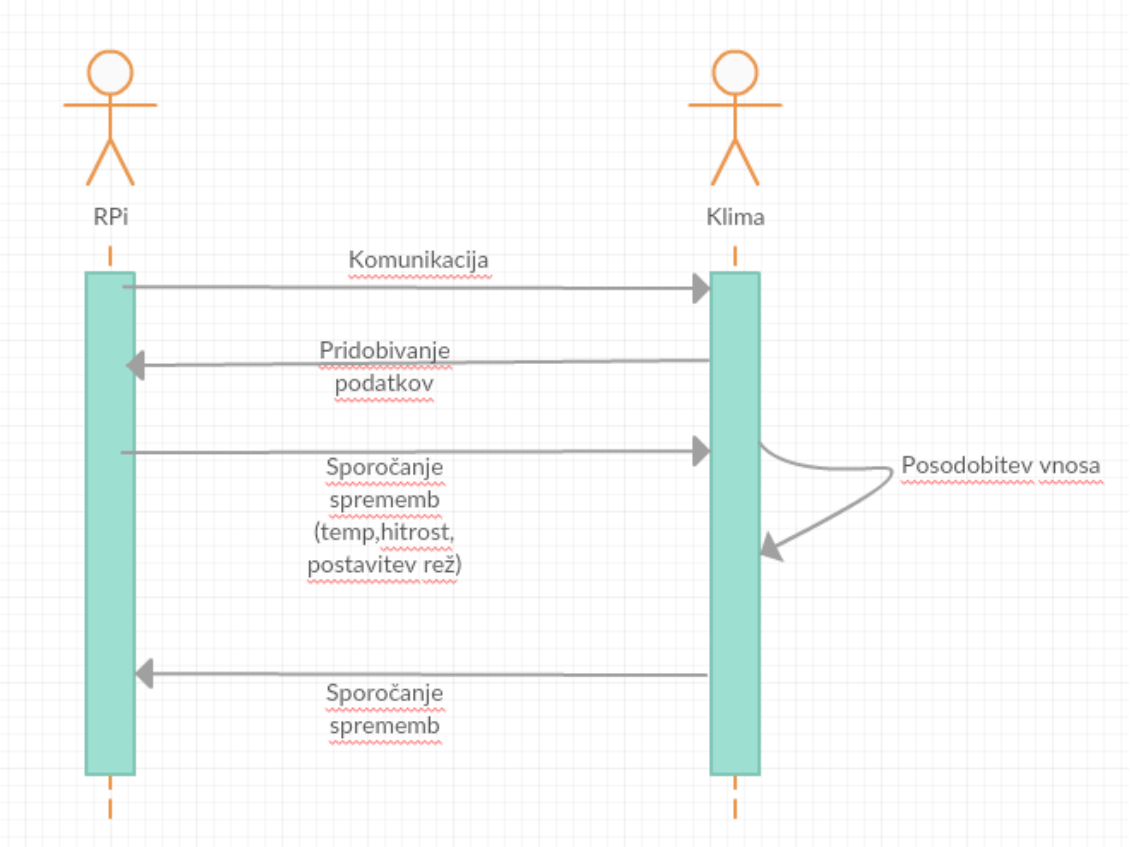
Prikazuje komunikacijo med RPi, TV in ojačevalcem (slika 4).



Slika 4 – diagram zaporedja za AV TV modul

3.3 Diagram klime

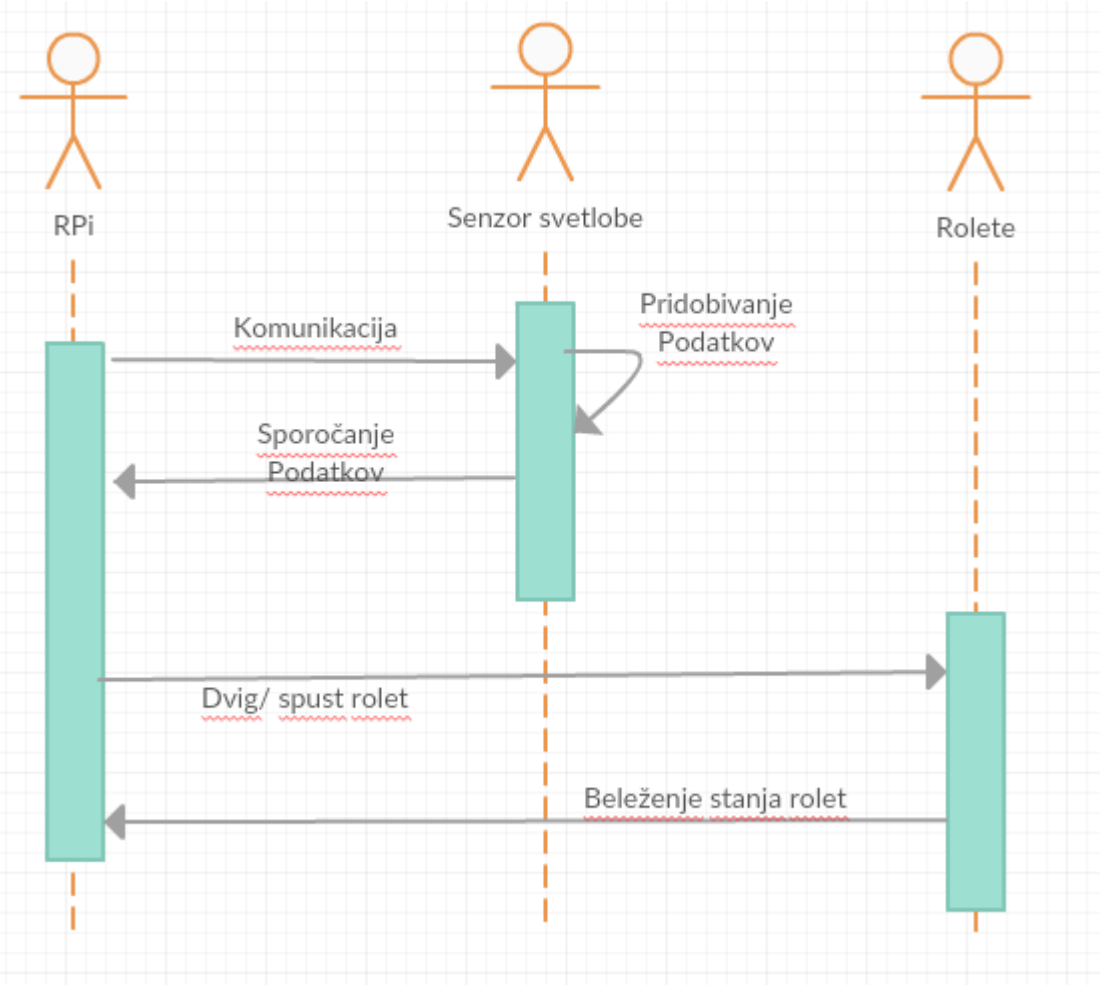
RPi sporoča klimi temperaturo, hitrost pihanja, postavitev rež (glede na vnos uporabnika), klima ta vnos izvrši in spremembo potrdi RPi (slika 5).



Slika 5 – Diagram prikazuje komunikacijo med klimo in RPi

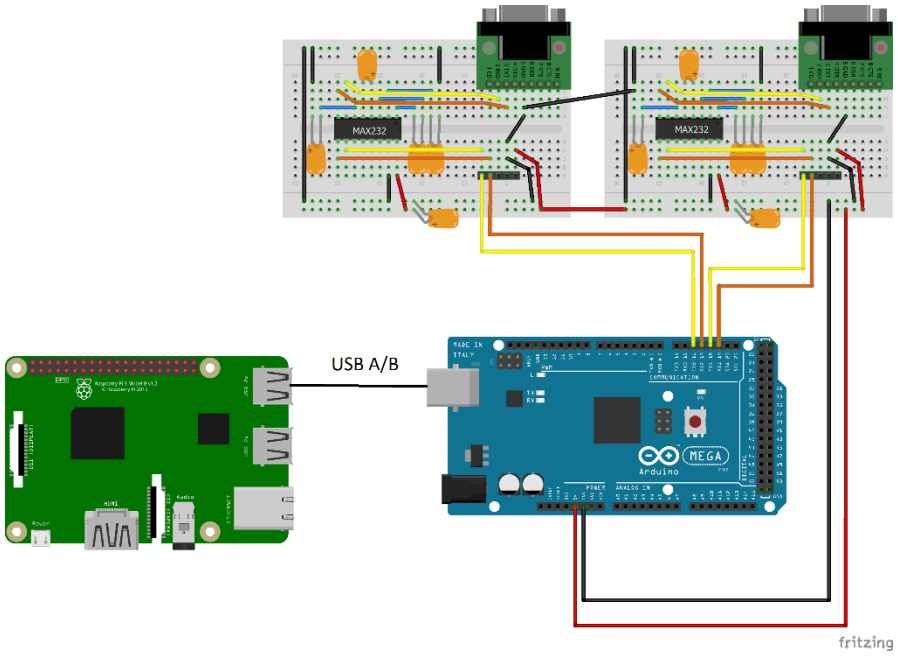
3.4 Diagram modula z rolete

RPi komunicira z modulom rolet in pa senzorjem za svetilnost. Najprej se pozanima kakšna je svetilnost in glede na dobljene podatke rolete dvigne ali spusti (slika 6).



Slika 6 – diagram modula za rolete

4. Shematski prikaz vezave Rpi in Arduino



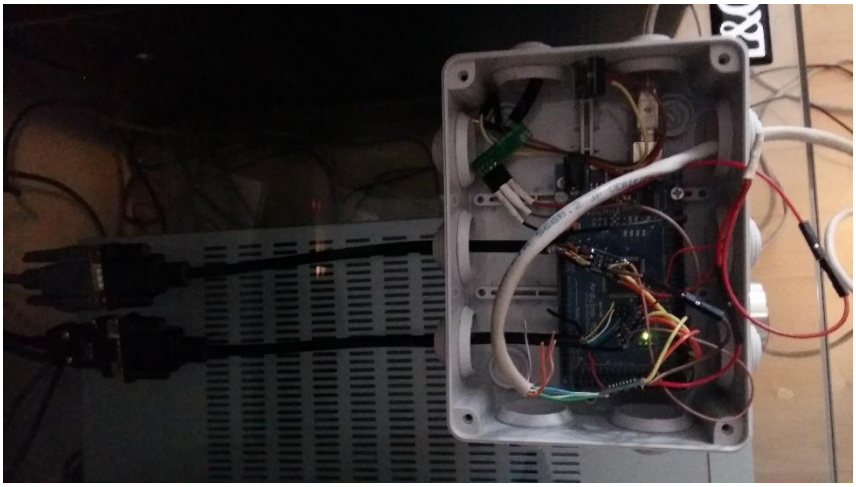
Slika 7 – Shema vezave

1x Raspberry Pi

1x Arduino Mega 2560

2x [MAX232 converter]

Slika 8 – MAX232 converter



Slika 9 - Praktična izvedba

5. Protokol za komunikacijo med Rpi in Arduinom

'H','U', 0x01, 0x01, 0x01, 0x0B, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF

Header : Message Type : Node ID : Device ID : Device Type : Payload Lenght: Pay Load : Check Sum

Header: je vedno 'H' kar označuje začetek bloka.

Message Type:

Tipi G in S sporočila podatki grejo v smeriu od RPi k Arduino plošči

G - Get state – zahteva po branju določenega NodeID: Device ID: Device Type

S - Set state – zahteva po pisanju v dololčen NodeID: Device ID: Device Type

Tip U sporočila, podatki grejo v smeri od Arduino plošče k Rpi.

U - Update state – to sporočilo inicijra Arduino plošča.

Device ID:

Označuje številko naprave. Primer, če imamo več temperaturnih senzorjev na Node ID 1, z Device ID naslavljamo prvi, drugi ali N-ti temperaturni senzor.

Device Type:

To polje označuje tip naprave in je označeno s število 0 – 255. Tipi naprav so navedeni v tabeli spodaj, lahko jo razširimo poljubno glede na lastne potrebe.

1 Temp

2 IR RX

3 IR TX

4 Humidity

5 Power Consumption

6 Water Counter

Payload Lenght:

Označuje dolžino v byte-ih sporočila. Max dolžina je 255 byte-ov.

Payload:

Vsebina sporočila. Format sporočila je poljuben in interpretacija sporočila je odvisna od naprave »Device Type«

Check Sum:

Cheksum polje, ki je seštevek vseh byteov od 'H' do vključno predzadnjega byta v spodnjem primeru do 0x34. Cheksum se računa po

'H','U', 0x01, 0x01, 0x01, 0x0B, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x34, 0xFF

Header : Message Type : Node ID : Device ID : Device Type : Payload Lenght: Pay Load : Check Sum

6. Dodajanje novih funkcionalnosti

V osnovi je potrebno dodati kodo na treh različnih koncih in sicer v:

- Arduino module
- Node.js datoteko sb-admin.js
- Spletno stran ki se nahaja pod /sb-admin/pages/

6.1 Primer dodajanja funkcionalnosti v Arduino modul

```
void setup() {
  // Nastavimo serijsko komunikacijo z Rpi
  Serial.begin(115200);
}

void loop() {

        // Preberemo nek senzor recimo temperaturni
          ReadTemperature(inData1);

        // Pošljemo podatek formatiran po našemu protokolu od Arduina proti  Raspberry Pi
            DomotixNetO('u',1,2,1,inData1);

        // Če je podatek z Raspberry Pi na voljo ga preberemo
            if ( Serial.available() ){
               inByte = Serial.read();
               inData += inByte;
               if (inByte == 'z'){
                   DomotixNetI(inData); // extract Payload from protocol
               inData = ""; // Clear recieved buffer
       }
  }


}
```

6.2 Dodajanje funkionalnosti v Node.js

Dodajanje funkcionalnosti v Node.js je precej poenostavljeno imamo dva switch stavka, eden je za spremljanje dogodkov ki ji prejemamo po websocket povezavi s spletne strani, drugi pa za dogodke, ki jih prejmemo z Arduino plošče.Del kode sb-admi.js kjer dodajamo dodatne naprave:

```
function DomotixNetI(dta){
 //console.log("dta: "+dta);
 var PayLo ="";
 if ((dta[0]=0x68) && (dta[1]=0x75)) {

   var swc = (DecVal(dta[6],dta[7]));
   console.log("Device Type: "+swc);
   switch(swc) {
/*      case 1:
         for (var d = 6; d < dta.length; d++) {
           PayLo += String.fromCharCode(dta[d]);
         }
         console.log("PayLo: "+PayLo,PayLo.length);
         valu=PayLo;
         io.emit('update', {label: 'LivingTemp' ,value: valu.toString()});
         break; */
      case 2:
        PayLo = "";
         for (var d = 10; d < dta.length; d=d+2) {
           PayLo += String.fromCharCode(DecVal(dta[d],dta[d+1]));
         }
         console.log("Case2 PayLo: "+PayLo);
         val2=PayLo.slice(5,PayLo.length-3);
         lbl=PayLo.slice(1,5)
         io.emit('update', {label: lbl.toString() ,value: val2.toString()});
         console.log("Marantz "+lbl,val2);
         break;
      case 3:
         PayLo = "";
         for (var d = 10; d < dta.length-5; d=d+2) {
           PayLo += String.fromCharCode(DecVal(dta[d],dta[d+1]));
         }
         console.log("Case3 PayLo: "+PayLo);
         var decval = DecVal(dta[8],dta[9]);
         console.log("DevicID: "+decval);
         if (decval == 195){
           var val3 = PayLo.split(":");
           temp195=val3[0].toString();
           hum195=val3[1].toString();
           console.log("val: "+val3);
           io.emit('update', {label: "temp195" ,value: temp195});
           io.emit('update', {label: "hum195" ,value: hum195});

         }
         if (decval == 187){
           var val4 = PayLo.split(":");
           console.log("val4: "+val4);
           temp187=val4[0].toString();
           hum187=val4[1].toString();
           io.emit('update', {label: "temp187" ,value: temp187});
           io.emit('update', {label: "hum187" ,value: hum187});
         }
         break;
      case 4: //določi poljubno številko Device ID
      // tukaj dodaj kodo za svojo funkionalnost
```

```
     break;

   }
 }
}
```

Del kode sb-admin.js, ki sprejema dogodke s spletne strani:

```
io.sockets.on('connection', function (socket) {
      sendTime();
      io.sockets.emit('initialize');

    // When the server receives a .message. type signal from the client
   socket.on('message', function (message) {

    console.log('Client message: ' + message);

    switch(message) {

     case "AC_ON":
        console.log('Case: AC_ON');
        DomotixNetO('s',1,4,1,"AC_ON");
        break;
     case "AC_OFF":
        console.log('Case: AC_OFF');
        DomotixNetO('s',1,4,1,"AC_OFF");
        break;
     case "AC_HEAT":
        console.log('Case: AC_HEAT');
        DomotixNetO('s',1,4,1,"AC_HEAT");
        break;
     case "AC_COOL":
        console.log('Case: AC_COOL');
        DomotixNetO('s',1,4,1,"AC_COOL");
        break;
     case "AC_DRY":
        console.log('Case: AC_DRY');
        DomotixNetO('s',1,4,1,"AC_DRY");
        break;
     case "IME_VAŠEGA_DOGODKA":
        console.log('Case: IME_VAŠEGA_DOGODKA ');
        DomotixNetO('s',1,4,1," IME_VAŠEGA_DOGODKA ");
        break;
```

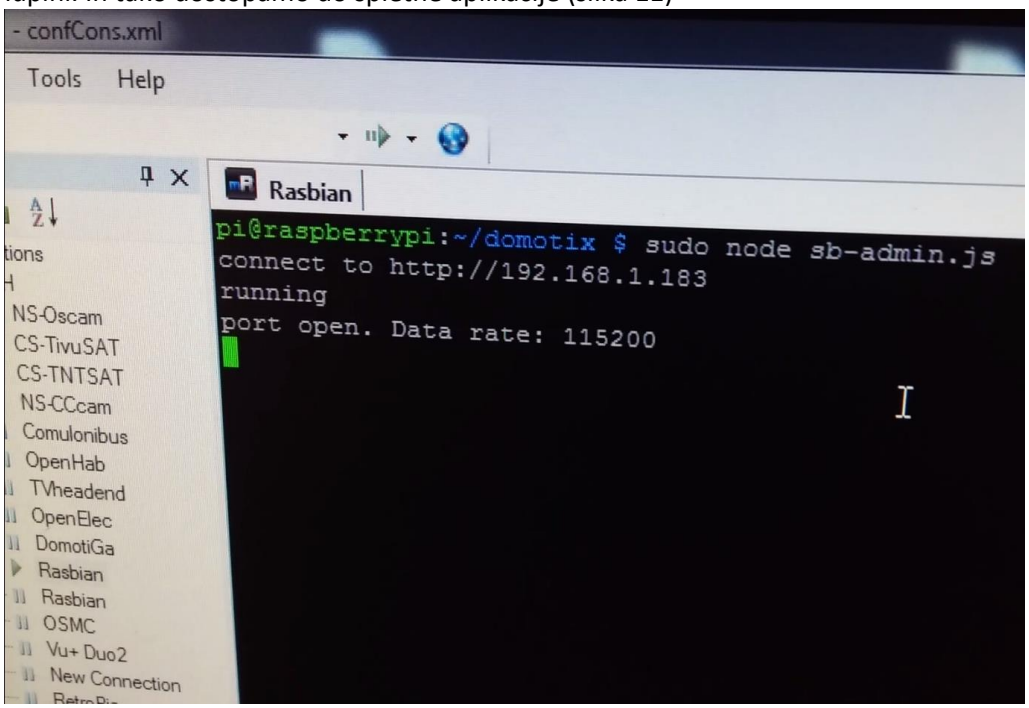
Del Java script kode spletnega uporabniškega vmesnika bathroom.html

```
                              <h4>Volume</h4>
<div id="MSV:">--</div>
                              <p>
                                  <button
onclick="javascript:OnClick('IME_VAŠEGA_DOGODKA');" type="button" class="btn
btn-default" style="width:80px">+</button>
                                  <button
onclick="javascript:OnClick('MSV:2');" type="button" class="btn btn-default"
style="width:80px">-</button>
```

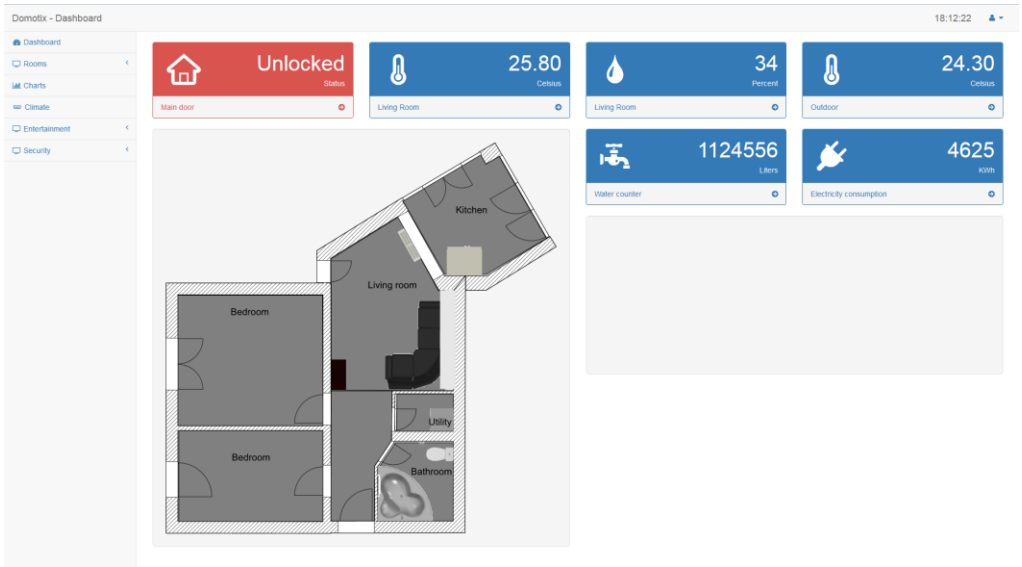
```
                                <button
onclick="javascript:OnClick('MSM:2');" type="button" class="btn btn-default"
style="width:80px">Mute On</button>
                                <button
onclick="javascript:OnClick('MSM:1');" type="button" class="btn btn-default"
style="width:80px">Mute Off</button>
                                </p>
                                <br>
                            </div>
                            <!-- /.panel-body -->
                        </div>
                        <!-- /.panel-default  -->
                    </div>
                    <!-- /.col-lg-3  -->
                </div>
                <!-- /.row -->
            </div>
            <!-- /#page-wrapper -->
            </div>
            <!-- /#wrapper -->

           <script src='/socket.io/socket.io.js'></script>
<script>
                    var socket = io();
                    socket.on('time', function(data) {
                        document.getElementById("time").innerHTML = data.time;
                    });
                    socket.on('initialize', function(data) {
                        socket.emit('message','MSP:?' );
                        socket.emit('message','MSC:?' );
                        socket.emit('message','MSV:?' );

                    });
                    socket.on('update', function(data) {
                        document.getElementById(data.label).innerHTML = data.value;
                    });
                    socket.on('error', console.error.bind(console));
                    socket.on('message', console.log.bind(console));

                </script>
```

7. WEB komunikacija

Centralna enota (RPi) se zažene in vzpostavi server preko katerega dostopaš z ip naslovom. Z ukazom sudo node sb-admin.js v Rasbian OS (slika 10). V brskalnik vnesemo ip naslov ki je prikazan v ukazni lupini. In tako dostopamo do spletne aplikacije (slika 11)



Slika 10



Slika 11

8. Literatura

1. Sook-Ling Chua, Lee Kien Foo, Sensor Selection in Smart Homes, pridobljeno aprila 2017, http://www.sciencedirect.com/science/article/pii/S1877050915031762
2. Jérémy Lapalu , Kevin Bouchard , Abdenour Bouzouane , Bruno Bouchard , Sylvain Giroux, 2013, Unsupeervised Mining of Activities for Smart Home Prediction, pridobljeno aprila 2017, http://www.sciencedirect.com/science/article/pii/S1877050913006753
3. https://www.npmjs.com/package/smart-home