CS2310 Project Report

Zinan Zhang | ziz22@pitt.edu
Dec 8 2016

Introduction

In this project, I implemented a Thermometer System, which is used for take care of human's body temperature. It is extended from exercise 4 and added lots of new features. Once it detects an abnormal temperature, it will send the temperature data to the monitor and then evaluate the data. The monitor will make a decision whether send a message to other component or not.

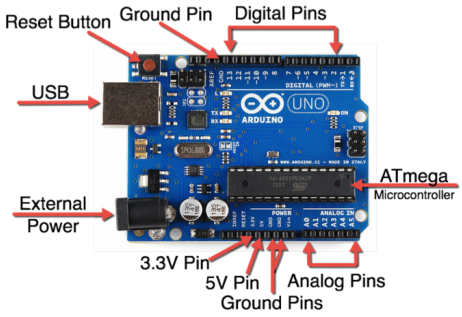
The object of this system is senior people living alone, especially for those senior people who not familiar with the modern telecom devices.

Background

- Design pattern
  The design of this system is based in the theory of patterns. Each pattern is a three-part rule, which expresses a relation between a certain context, a problem, and a solution. So when we meet with a problem in a certain context, we can then generate a solution to solve the problem. In other words, the pattern is an attempt to discover some invariant feature that distinguishes good places from bad places with respect to some particular system of forces (constraints).

- System user
  The user of this system mostly is senior people who live alone. As the their kids grows up and themselves get aged, daily life will become much more difficult. Some of them may lose their hearing; some may lose their sight. Lots of potential dangerous will increase the difficulty for senior people to keep healthy. However, with this Thermometer System, they can easily get in touch with the health care center at any time.

- Arduino
  Arduino is an open source hardware and software project first introduced in 2005 based on 8-bit Atmel AVR, aiming to provide an accessible way for novices and professionals to create devices that interact with their environment using sensors and actuators.

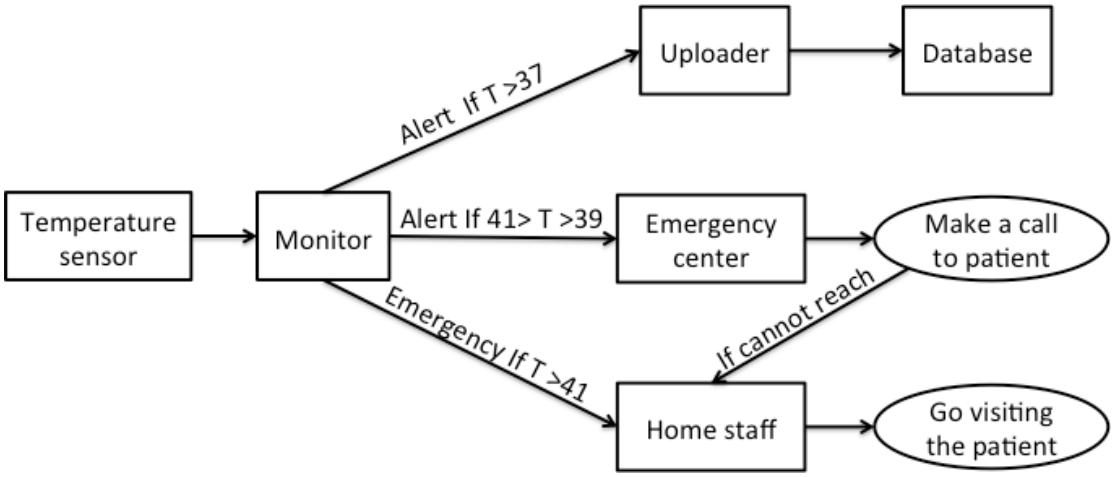


Arduino

Temperature Sensor

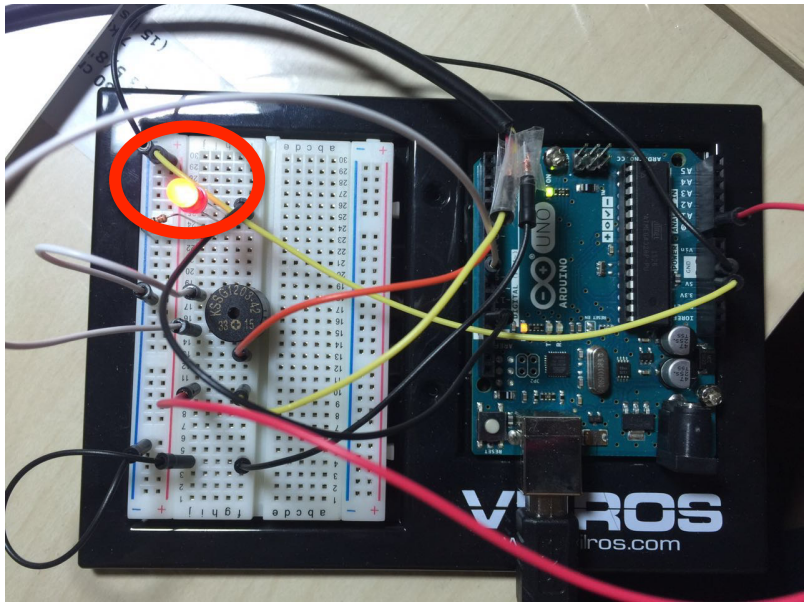
System Structure

The system is consisted of five components:
Temperature sensor, Monitor, Uploader, Emergency Center, Home Staff.

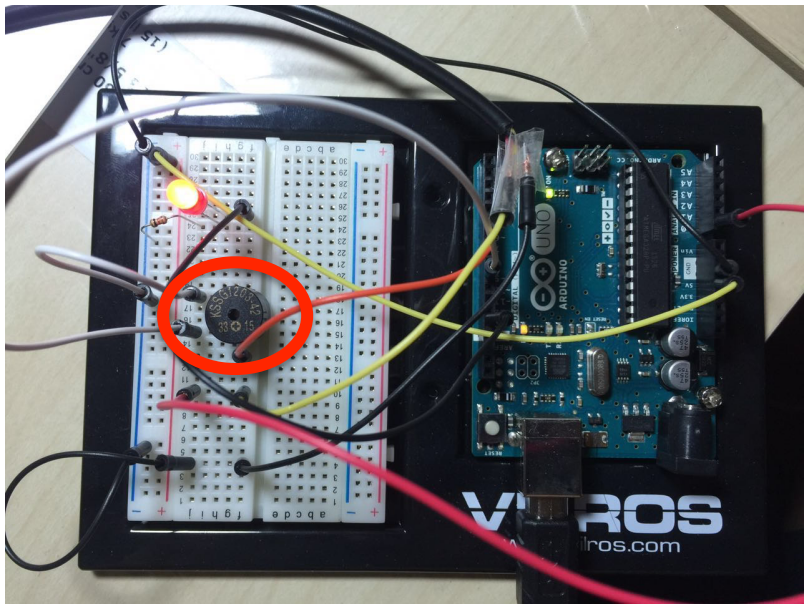


Component	Function
Temperature sensor	Detecting the human body's temperature.
Monitor	Deciding to move forward the data to which component.
Uploader	Uploading the data to database and sending an email back to the patients, telling them that they have an abnormal body temperature.
Emergency Center	Making a call to patient to have a quick diagnose and giving some advise. If they cannot reach the patient, then send an emergency to Home Staff.
Home Staff	Going to visit the patient with ambulance.

As we can see from the diagram from above, the temperature sensor can detect human body's temperature in order to take care of their health. When the sensor gets a temperature, it will send the data to Monitor. The sensor can not only send the data to the monitor, but also can give some alert or feed back to the patient immediately. There is a red light and a small buzzer on the device. Once the patient's body temperature is abnormal high, both of the red light and buzzer will start working to alert the patient.

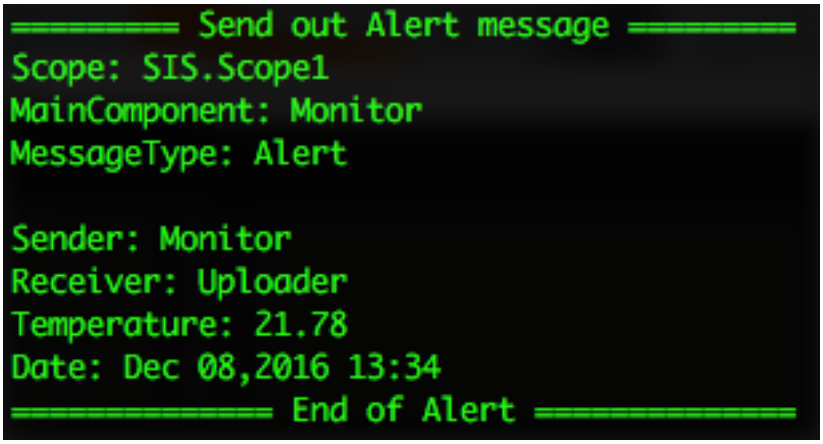


The red light is for senior people who lose their hearing. (###extra deeds)

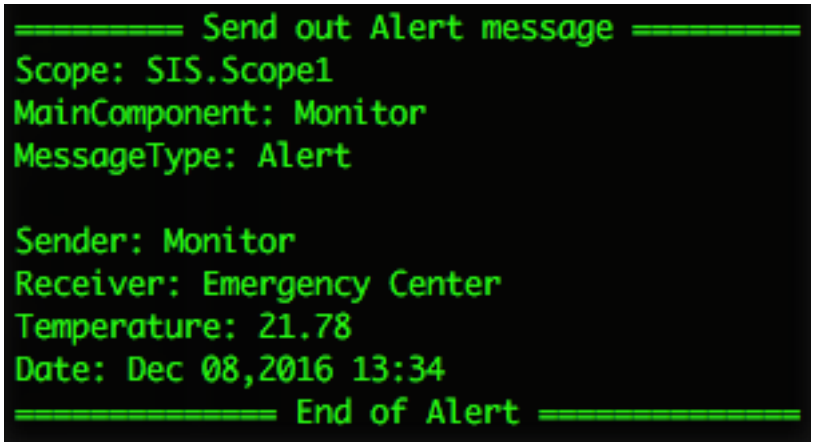


The buzzer is for senior people who lose their sight. (###extra deeds)

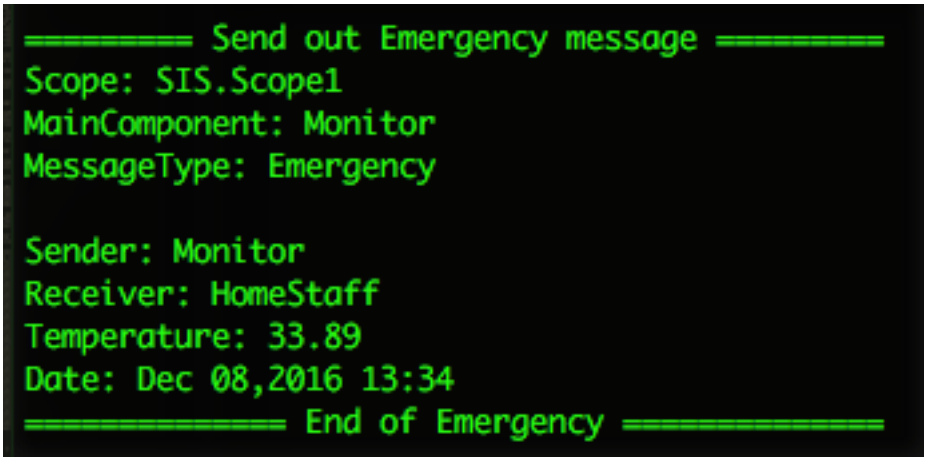
Then the Monitor will decide to upload he data, send alert to Emergency Center or send emergency to Home Staff. If the data is abnormal but still in the normal range (like a temperature between 37 and 39), the monitor will send the data to the uploader to upload it and send an alert to the emergency center as well.



Send data to the uploader

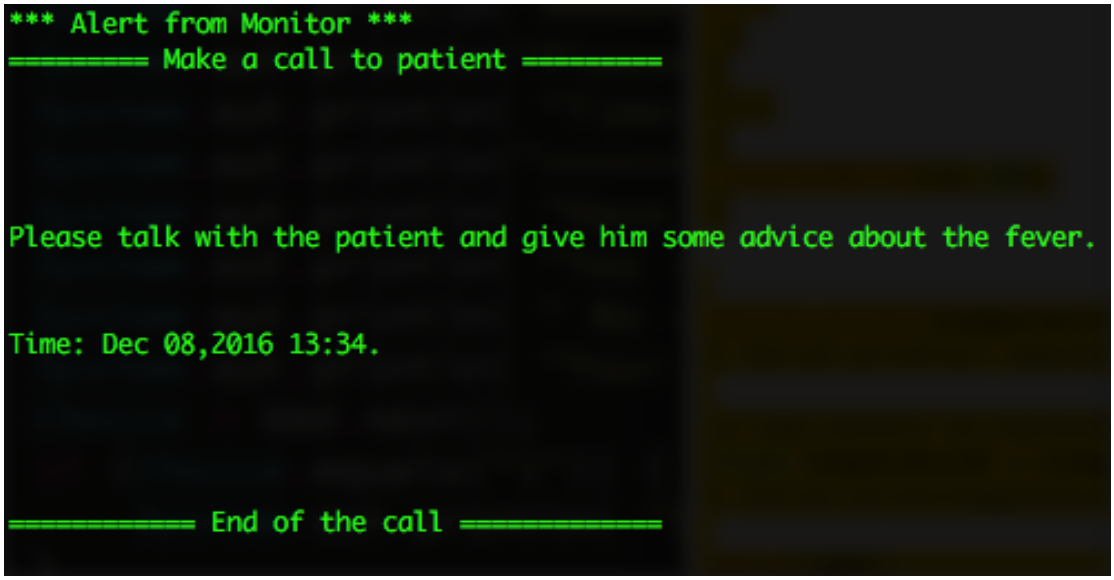


Send alert message to the Emergency Center

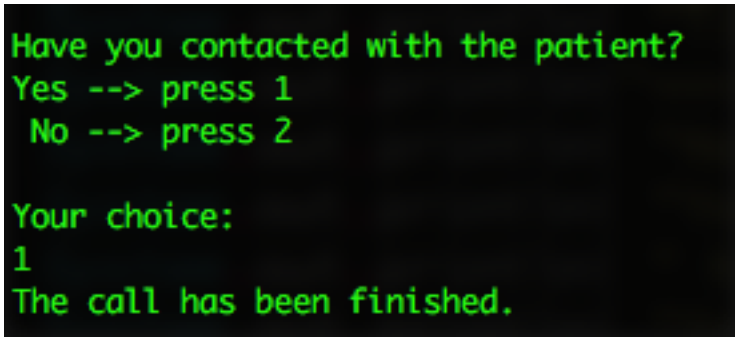


Send emergency message to the Home Staff

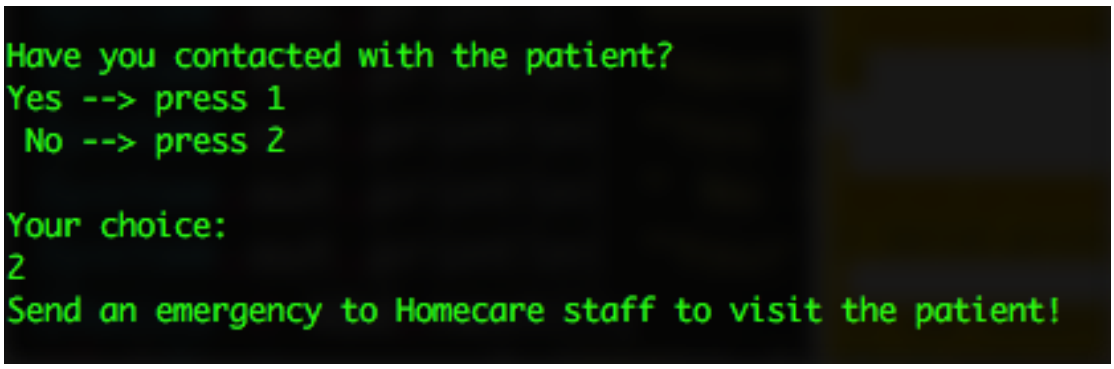
The alert is for telling the emergency center staffs to make a phone call to the patient. At this time, we assume that the patient just get a fever and not a big deal. But his/her doctor still needs to have a talk with him/her in order to make sure their fine. However, once the doctor cannot get in touch with the patient, they have to tell the Home Staffs to go visit the patient in case they are in danger. Just one press by the doctor will finish this procedure. I implemented a simple GUI here. (###extra deeds)



Receive the alert message from the monitor.

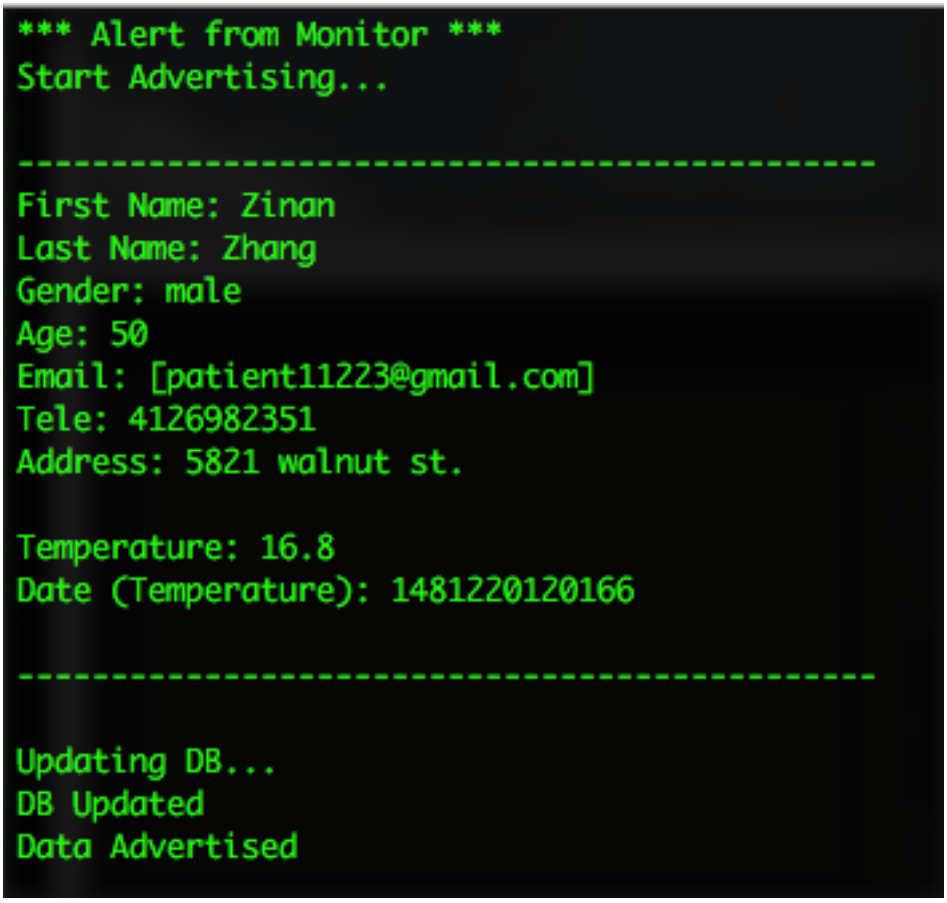


GUI for deal with making a phone call.



GUI for deal with making a phone call.

If the data received by the monitor is extremely abnormal (like a temperature higher than 40), the monitor will send the data to the uploader and send an emergency message to the Home Staff at once. And the Home Staff will go visit the patient without any hesitate, because the patient may have an emergency situation or their house in firing.



Receive data from monitor and upload data to database.

Extra deeds

- Red light feed back
- Buzzer feed back
- Emergency Center GUI

Code

1. For Arduino part

```
//TMP36 Pin Variables
int sensorPin = 0; //the analog pin the TMP36's Vout (sense) pin is connected to
                                    //the resolution is 10 mV / degree centigrade with a
                                    //500 mV offset to allow for negative temperatures
// TONES  ==========================================
// Start by defining the relationship between
//          note, period, &    frequency.
#define   c         3830       // 261 Hz
#define   d         3400       // 294 Hz
#define   e         3038       // 329 Hz
#define   f         2864       // 349 Hz
#define   g         2550       // 392 Hz
#define   a         2272       // 440 Hz
#define   b         2028       // 493 Hz
#define   C         1912       // 523 Hz
// Define a special note, 'R', to represent a rest
#define   R         0
// SETUP ============================================
// Set up speaker on a PWM pin (digital 9, 10 or 11)
int speakerOut = 9;

void setup()
{
   Serial.begin(9600);    //Start the serial connection with the computer
                                    //to view the result open the serial monitor
   pinMode(13,OUTPUT);

   pinMode(speakerOut, OUTPUT);
//    if (DEBUG) {
//       Serial.begin(9600);
//    }
}

// MELODY and TIMING  =======================================
//    melody[] is an array of notes, accompanied by beats[],
//    which sets each note's relative length (higher #, longer note)
int melody[] = {    C,    b,    g,    C,    b,      e,    R,    C,    c,    g, a, C };
int beats[]    = { 16, 16, 16,    8,    8,    16, 32, 16, 16, 16, 8, 8 };
int MAX_COUNT = sizeof(melody) / 2; // Melody length, for looping.
```

```
// Set overall tempo
long tempo = 10000;
// Set length of pause between notes
int pause = 1000;
// Loop variable to increase Rest length
int rest_count = 100;
// Initialize core variables
int tone_ = 0;
int beat = 0;
long duration  = 0;

// PLAY TONE  ==============================================
// Pulse the speaker to play a tone for a particular duration
void playTone() {
  long elapsed_time = 0;
  if (tone_ > 0) { // if this isn't a Rest beat, while the tone has
    //  played less long than 'duration', pulse speaker HIGH and LOW
    while (elapsed_time < duration) {

      digitalWrite(speakerOut,HIGH);
      delayMicroseconds(tone_ / 2);

      // DOWN
      digitalWrite(speakerOut, LOW);
      delayMicroseconds(tone_ / 2);

      // Keep track of how long we pulsed
      elapsed_time += (tone_);
    }
  }
  else { // Rest beat; loop times delay
    for (int j = 0; j < rest_count; j++) { // See NOTE on rest_count
      delayMicroseconds(duration);
    }
  }

}

void loop()                                // run over and over again
{
 //getting the voltage reading from the temperature sensor
 int reading = analogRead(sensorPin);

float voltage = reading * 5.0;
```

```
  voltage /= 1024.0;

  // print out the voltage
// Serial.print(voltage); Serial.println(" volts");

  // now print out the temperature
  float temperatureC = (voltage - 0.5) * 100 ;
//converting from 10 mv per degree wit 500 mV offset
//to degrees ((voltage - 500mV) times 100)

if (temperatureC > 39)
  {
    digitalWrite(13,HIGH); //alert light
    //Alert Sound
    for (int i=0; i<MAX_COUNT; i++) {
        tone_ = melody[i];
        beat = beats[i];

        duration = beat * tempo; // Set up timing

        playTone();
        // A pause between notes...
        delayMicroseconds(pause);

        if (DEBUG) { // If debugging, report loop, tone, beat, and duration
        }
    }
  }
  else
  {
    digitalWrite(13,LOW);
  }

  Serial.println(temperatureC);
// Serial.println(" degrees C");

  // now convert to Fahrenheit
  float temperatureF = (temperatureC * 9.0 / 5.0) + 32.0;
// Serial.print(temperatureF); Serial.println(" degrees F");

  delay(1000); //waiting a second
}
```

2. For Monitor conditional statement

```
case "Temp":
        String tem = kvList.getValue("Temp");
        String datTe = kvList.getValue("Date");

        if (tem != null && !tem.equals(""))
        {
                reading.temp = Double.parseDouble(tem);
        }
        if (datTe != null && !datTe.equals(""))
        {
                reading.dateTemp = Long.parseLong(datTe);
        }
        KeyValueList alert = new KeyValueList();
        SimpleDateFormat sdf = new SimpleDateFormat("MMM dd,yyyy HH:mm");
        Date resultdate = new Date(reading.dateTemp);
        if (reading.temp > 37) {
                        System.out.println("======== Send out Alert message ========");
                        alert.putPair("Scope", SCOPE);
                        alert.putPair("MainComponent", "Uploader");
                        alert.putPair("MessageType", "Alert");
                        alert.putPair("Sender", "Monitor");
                        alert.putPair("Receiver", "Uploader");
                        alert.putPair("Temp", Double.toString(reading.temp));
                        alert.putPair("Date", Long.toString(reading.dateTemp));

                        StringBuilder messg = new StringBuilder();
                        messg.append("Scope: " + SCOPE + "\n");
                        messg.append("MainComponent: " + "Monitor" + "\n");
                        messg.append("MessageType: " + "Alert" + "\n\n");
                        messg.append("Sender: " + "Monitor" + "\n");
                        messg.append("Receiver: " + "Uploader" + "\n");
                        messg.append("Temperature:  "  +  Double.toString(reading.temp)  +
"\n");
                        messg.append("Date: " + sdf.format(resultdate) + "\n");
                        messg.append("============= End of Alert ============\n");
                        System.out.println(messg);
                        encoder.sendMsg(alert);
        }

        if (reading.temp > 39 && reading.temp < 41) {
                System.out.println("========= Send out Alert message =========");

                alert.putPair("Scope", SCOPE);
```

```
        alert.putPair("MainComponent", "Monitor");
        alert.putPair("MessageType", "Alert");
        alert.putPair("Sender", "Monitor");
        alert.putPair("Receiver", "EmergencyCenter");
        alert.putPair("Temp", Double.toString(reading.temp));
        alert.putPair("Date", Long.toString(reading.dateTemp));

        StringBuilder messg = new StringBuilder();
        messg.append("Scope: " + SCOPE + "\n");
        messg.append("MainComponent: " + "Monitor" + "\n");
        messg.append("MessageType: " + "Alert" + "\n\n");
        messg.append("Sender: " + "Monitor" + "\n");
        messg.append("Receiver: " + "Emergency Center" + "\n");
        messg.append("Temperature: " + Double.toString(reading.temp) + "\n");
        messg.append("Date: " + sdf.format(resultdate) + "\n");
        messg.append("============== End of Alert ==============\n");
        System.out.println(messg);
        encoder.sendMsg(alert);
}

if (reading.temp > 41) {
        System.out.println("======= Send out Emergency message ========");

        alert.putPair("Scope", SCOPE);
        alert.putPair("MainComponent", "Monitor");
        alert.putPair("MessageType", "Emergency");
        alert.putPair("HelperComponents", "HomeStaff");
        alert.putPair("Note", "Go visiting the patient RIGHT! NOW!");
        alert.putPair("Sender", "Monitor");
        alert.putPair("Receiver", "HomeStaff");
        alert.putPair("Temp", Double.toString(reading.temp));
        alert.putPair("Date", Long.toString(reading.dateTemp));

        StringBuilder messg = new StringBuilder();
        messg.append("Scope: " + SCOPE + "\n");
        messg.append("MainComponent: " + "Monitor" + "\n");
        messg.append("MessageType: " + "Emergency" + "\n\n");
        messg.append("Sender: " + "Monitor" + "\n");
        messg.append("Receiver: " + "HomeStaff" + "\n");
        messg.append("Temperature: " + Double.toString(reading.temp) + "\n");
        messg.append("Date: " + sdf.format(resultdate) + "\n");
        messg.append("============End of Emergency ============\n");
        System.out.println(messg);
        encoder.sendMsg(alert);
```

```
        }
        break;
```

3. For emergency center conditional statement

```
case "Monitor":
        String spo = kvList.getValue("Monitor");
        String datMO = kvList.getValue("Date");
        Scanner kbd = new Scanner(System.in);
        String Choice;

        if (spo != null && !spo.equals(""))
        {
            reading.spo2 = Integer.parseInt(spo);
        }
        if (datMO != null && !datMO.equals(""))
        {
            reading.dateSPO2 = Long.parseLong(datMO);
        }
        long yourmilliseconds = System.currentTimeMillis();
        SimpleDateFormat sdf = new SimpleDateFormat("MMM dd,yyyy HH:mm");
        Date resultdate = new Date(yourmilliseconds);
        System.out.println("========= Make a call to patient =========\n\n\n\n");
        System.out.println("Please talk with the patient and give him some advice about
the fever.\n\n");
        System.out.println( "Time: " + sdf.format(resultdate) + ".\n\n\n\n");
        System.out.println("============ End of the call
==========\n\n\n\n");
        System.out.println( "Have you contacted with the patient?");
        System.out.println( "Yes --> press 1");
        System.out.println( " No --> press 2\n");
        System.out.println( "Your choice: ");
        Choice = kbd.next();
        if (Choice.equals("1")) {
            System.out.println( "The call has been finished.\n");
        }
        else{
            System.out.println( "Send an emergency to Homecare staff to visit the
patient!\n");
        }
        break;
```

4. For Home Staff conditional statement

```
case "Emergency":
        String sup = kvList.getValue("MainComponent");
```

```
        String auxs = kvList.getValue("HelperComponents");
        String note = kvList.getValue("Note");

        System.out.println("\n*** Emergency Alert from " + sup + " backed by " +
auxs+" ***");

        System.out.println("Start Advertising...\n");
        long yourmilliseconds = System.currentTimeMillis();
        SimpleDateFormat sdf = new SimpleDateFormat("MMM dd,yyyy HH:mm");
        Date resultdate = new Date(yourmilliseconds);
        System.out.println("Data Advertised\n");
        System.out.println("========= Visit the patient =========\n\n\n\n");
        System.out.println( note + ".\n");
        System.out.println( "Time: " + sdf.format(resultdate) + ".\n\n\n\n");
        System.out.println("================ End ================\n\n\n\n");
        break;
```

5. For collecting raw data from sensor (wrote by python)

```
import serial
import threading
import time
from datetime import datetime

if __name__ == '__main__':
   pass
   try:
      arduino = serial.Serial('/dev/cu.usbmodem1421', 9600)
      print arduino
      temp = arduino.readline()
      print temp
   except:
      print 'error'
      exit(0)
```