CS2310 Final Report

Personal Health care System

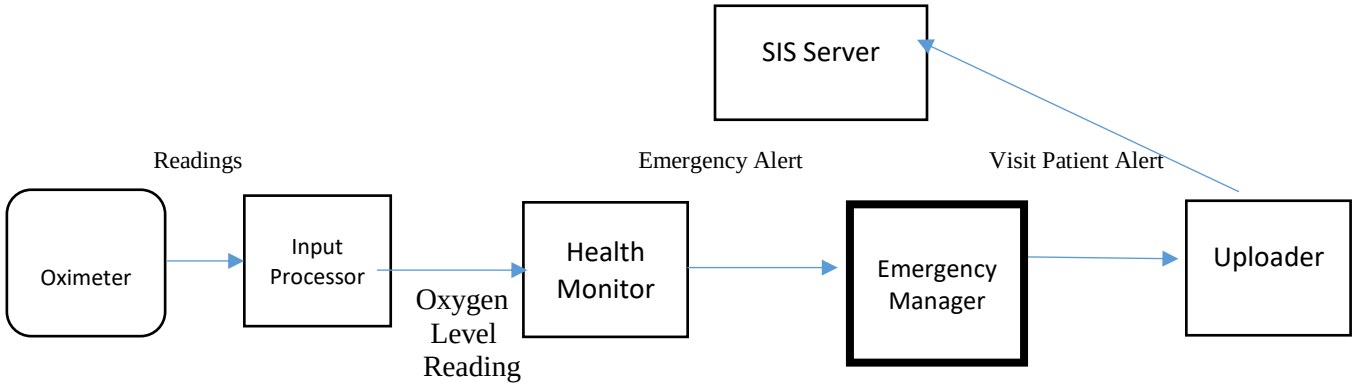
Manali Shimpi

Introduction:

This term project extends the exercise 4.In this project I will be using CMS 50D+ Blue Finger Pulse Oximeter to get the readings of oxygen levels in the human body.

There are 4 components in the system.

1. Input Processor
2. Oxygen Level Monitor
3. Emergency Manager(Super Component)
4. Uploader



The project is used to detect abnormal oxygen levels in a human body.

The following chart is used to check for specific conditions.

	Normal	**Hypoxia**	**Hyperoxia**
Oxygen Level	**90 - 100**	**< 90**	**>100**

Scenario:

The CMS 50D+ Blue Finger Oximeter takes the oxygen level readings in a patient. These readings are then forwarded to the input processor which refines the reading and also filters out invalid or any garbage values that may occur.

The input processor then forwards the processed readings to the Oxygen Level Monitor. According to the chart for oxygen levels and corresponding conditions the Monitor will send alert to the Emergency

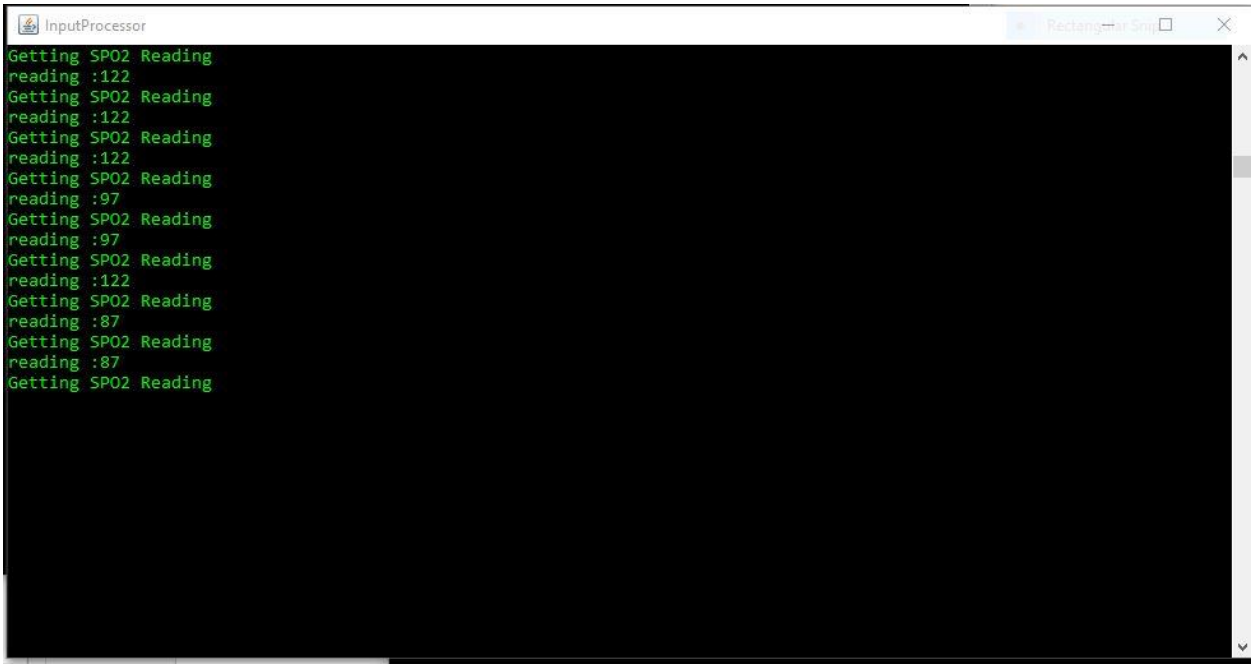
Manager. After receiving the alert the emergency Manager will send an emergency Email at chronobot@ksiresearch.org.

Components:

Input Processor:

The Input processor takes the input from the sensor and provides some filtering on the data provided by the sensor. It removes invalid data and any garbage data or readings that are found. After processing, the input processor sends this data to the Monitor.

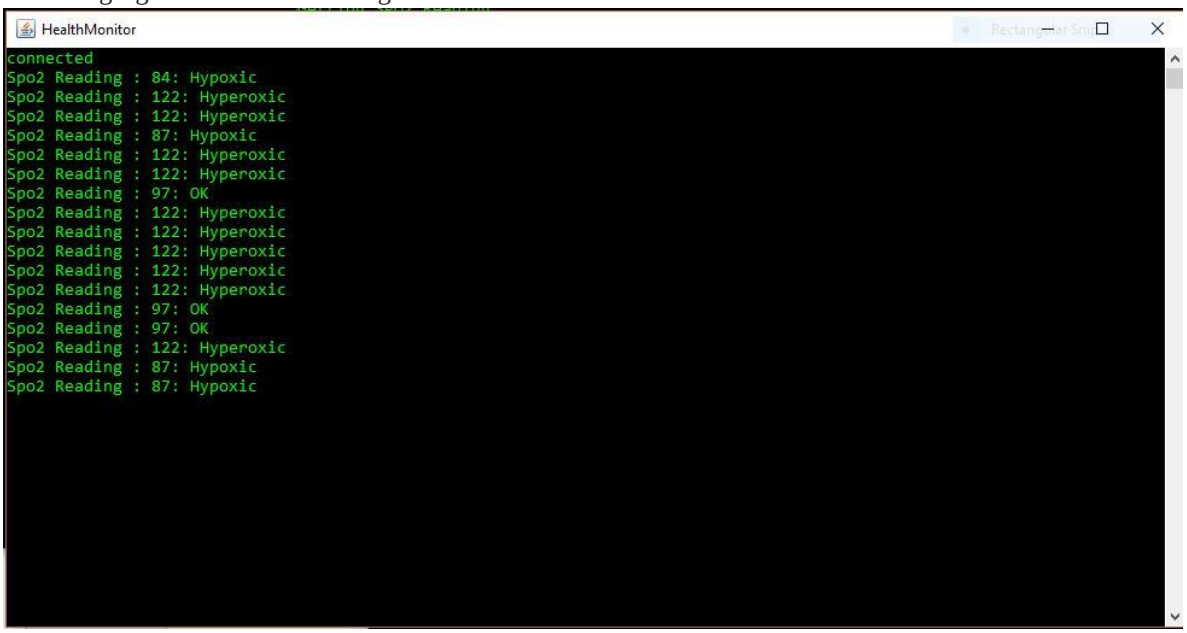
Following figure shows the working of input processor.



Health Monitor:

After receiving oxygen readings from the input processor, the health monitor checks the readings for the conditions of Hypoxia or Hyperoxia. The health monitor shows the readings along with the condition according to those readings as shown in the above chart. If the readings found to be not normal, the health monitor sends alert to the Emergency Manager.

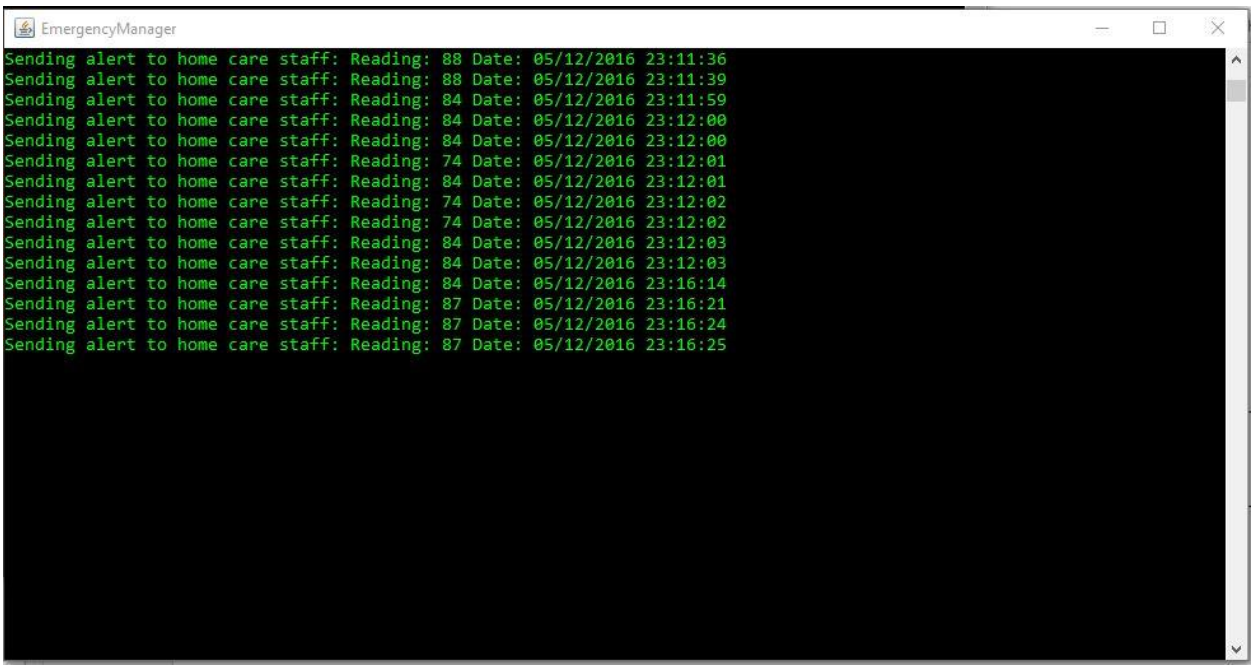
Following figure shows the working of Health Monitor.



Emergency Manager:

This is a super component. The Emergency Manager receives alert from the Health Monitor. This alert is shown on the screen of emergency Manager.

Following figure shows the working of Emergency Manager.



Uploader:

The uploader is a component of type, Advertiser. Uploader is basically used to email to the chronobot about the emergency case. Email consists of Name of the Patient, Address of the Patient and oxygen level reading.

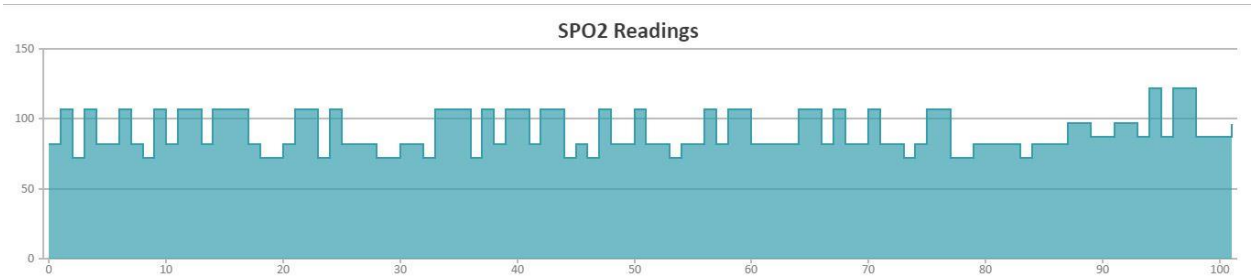
###Extra Deed:

Visualization of sensor Data using charts:

I have created visualization of the data in the form of charts as shown below.

The option to create a visualization or not is asked at the start of Input processor .If we enter “y”, a file will be created in html format. After running this file a chart can be seen on a browser.

Following figure shows how the chart is displayed.



###Generation of json Data:

I am also generating the data in json format and storing it in a file.

e.g.

{"Date": "06/12/2016 00:02:37", "SPO2": "99"}

Possible Future Work:

Currently, I am using only one sensor i.e. oximeter. More sensors can be incorporated such as to measure pulse, blood pressure, sugar level etc. Also the project can be expanded to provide analysis of the readings of various sensors and also patients’ profiles can be saved in the database. For example, if there are 4 members in the family, each member will have their profile stored in the system to generate personalized analysis of their health using sensors.

Appendix:

Code to get the values from sensor at serial port:

```
import java.io.BufferedWriter;
import java.io.DataOutputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStream;
import java.io.PrintWriter;
```

```
import java.net.Socket;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Date;
import java.util.Enumeration;
import java.util.List;
import java.util.Random;
import java.util.TooManyListenersException;

import javax.comm.CommPortIdentifier;
import javax.comm.PortInUseException;
import javax.comm.SerialPort;
import javax.comm.SerialPortEvent;
import javax.comm.SerialPortEventListener;
import javax.comm.UnsupportedCommOperationException;

public class SimpleRead implements Runnable, SerialPortEventListener {
        static CommPortIdentifier portId;
        static Enumeration portList;
        static int handle = 0,p=1,oldspo2=0;
        int skipper = 0;
        InputStream inputStream;
        SerialPort serialPort;
        Thread readThread;
        static PrintWriter pulseWriter;
        static DateFormat dateFormat = new SimpleDateFormat(("dd/MM/yyyy HH:mm:ss"));
        static  ArrayList<Integer> arrlist=new ArrayList<Integer>();
        Random rn=new Random();

        public static void main(String[] args) throws IOException {


                   // populate the list


                portList = CommPortIdentifier.getPortIdentifiers();

                while (portList.hasMoreElements()) {

                        portId = (CommPortIdentifier) portList.nextElement();
                        if (portId.getPortType() == CommPortIdentifier.PORT_SERIAL) {
                                if (portId.getName().equals("COM3")) {
                                        SimpleRead reader = new SimpleRead();
                                }
                        }

                }//while end
        }

        public SimpleRead() {
                try {
                        serialPort = (SerialPort) portId.open("SimpleReadApp", 2000);
                } catch (PortInUseException e) {
                        System.out.println(e);
                }
                try {
                        inputStream = serialPort.getInputStream();
                } catch (IOException e) {
                        System.out.println(e);
                }
                try {
                        serialPort.addEventListener(this);
                } catch (TooManyListenersException e) {
                        System.out.println(e);
                }
                serialPort.notifyOnDataAvailable(true);
                try {
                        serialPort.setSerialPortParams(19200, SerialPort.DATABITS_8,
SerialPort.STOPBITS_1, SerialPort.PARITY_NONE);
```

```
        } catch (UnsupportedCommOperationException e) {
                System.out.println(e);
        }
        readThread = new Thread(this);
        readThread.start();
}

public void run() {
        try {
                Thread.sleep(20000);
        } catch (InterruptedException e) {
                System.out.println(e);
        }
}

public void serialEvent(SerialPortEvent event) {

        switch (event.getEventType()) {
        case SerialPortEvent.BI:
        case SerialPortEvent.OE:
        case SerialPortEvent.FE:
        case SerialPortEvent.PE:
        case SerialPortEvent.CD:
        case SerialPortEvent.CTS:
        case SerialPortEvent.DSR:
        case SerialPortEvent.RI:
        case SerialPortEvent.OUTPUT_BUFFER_EMPTY:
                break;
        case SerialPortEvent.DATA_AVAILABLE:
                handle++;

                byte[] readBuffer = new byte[20];
                int ascii;
                int item,maxfreq=0,freq,curritem = 0;
                try {
                        while (inputStream.available() > 0) {

                                int numBytes = inputStream.read(readBuffer);
                        }
                        int index = 1;
                        Byte regular = readBuffer[index];

                        DateFormat dateFormat = new SimpleDateFormat(("dd/MM/yyyy"));
                        int spo2 = (int) regular;
                        System.out.println("before loop " +arrlist.size());

                                System.out.println("inside loop "+arrlist.size());
                                if(spo2>60)
                                arrlist.add(spo2);

                        if(arrlist.size()==15){
                         for(int i=0;i<arrlist.size();i++)
                         {
                                 item=arrlist.get(i);
                           freq = Collections.frequency(arrlist, arrlist.get(i));
                           if(freq>maxfreq)
                           {
                                   maxfreq=freq;
                                   curritem=item;
                           }

                         }
                         System.out.println("Max Freq is "+maxfreq+ " of " +curritem);
                          arrlist.clear();


                int x;

                if (maxfreq>10 )
                        {
                                p++;
```

```
                                oldspo2=spo2;
                        skipper++;

                        Socket socket = new Socket("127.0.0.1", 50011);
                        DataOutputStream d = new
DataOutputStream(socket.getOutputStream());
                // if(p%5==0)
                 //     spo2=spo2/2;
                        //x=rn.nextInt(10);
                        /*if(x>7)
                        {
                                curritem=curritem-10;

                        }
                        else if (x<4){
                                curritem=curritem+25;
                        }*/
                        d.writeUTF(curritem + "");
                        d.close();
                        socket.close();
                }
                        }
                }catch (IOException e) {
                        System.out.println(e);
                }



                break;
        }

    }
}
```