Team MDC Technology Feasibility, $V1.0$

Justin Poehnelt
Ahmad M. Meer
David Nagel
Jack Burrell

October 22, 2015

Contents

1 Introduction

The Mobile Data Collection project aims to create an extensible environment upon which numerous sensors and microcontroller devices can be used to collect and store data about an individual's gait cycle. The data must be accessible in near real-time through a centralized data repository.

1.1 Technologies

The technologies range from an embedded system all the way to a high performance write-heavy database. Many options exist for bridging these two ends of the project including the use of a Android device or Raspberry Pi and a modern webserver.

1.2 Document Summary

This document aims to clarify many of the risks associated with the objective and highlight some of the decisions that will allow the project to be feasible from a technological perspective.

2 Technology Overview

There are five primary layers to the project; the microcontroller and sensors, a mobile control node, a web server application, a database and a data analysis tool. Each of these layers must be able to reliably communicate a large amount of data and some of the layers must be able to do so in a potentially unpredictable physical environment.

2.1 Microcontrollers

The lowest level of the project likely involves the most risk due to the background of the team members. One of the primary decisions is the choice of microcontroller.

2.1.1 Teensy 3.2

The Teensy 3.2 is a complete USB-based microcontroller that is Arduino compatable. It has an ARM processor of 16 MHz and is 32 bit. It is very easy to push a program to with the Teensy Loader application. ultimately, the team decided to use the Teensy for the 32 bit ARM processor. Due to the nature of the project, the sensors and hardware on the feet need to provide as accurate as possible data. The 32 bit processor allows us to have a fast clock speed, and access more RAM memory. The Teensy does have its own drawbacks though. While is is relatively simple, there is a level of overhead that needs to be learned in order to program it. Also, unlike the Arduino UNO, it only comes as microcontroller, and will require some additional hardware support from the team.

2.1.2 Arduino UNO

The Arduino UNO is a micrcontroller board that allows for easy extensibility. The arduino UNO is not just a micrcontroller, it exists on a board that has some integrated peripherals that allow for easy use. The microcontroller that does exist on it however, is 16Mhz and based on 8 bit AVR architecture. While the Teensy will be used over the Arduino UNO, the Arduino still exists as a platform from which easy testing can be done of various sensors, due to its supported hardware.

2.2 Microcontroller Data Transfer

2.2.1 Overview

The next logical challenge after selecting the microcontroller is the determining the method by which data will be transferred from the microcontroller to a control node. This decision hinges upon the choice of control node and the two cannot easily be separated.

2.2.2 Bluetooth

Bluetooth 2.0 is a feasible technology for transferring data from the microcontroller to its control node. It allows for asynchronous transmission and reception of data. Most importantly, it is widely supported by more powerful devices such as smartphones. Ultimately Bluetooth was decided upon by the team due to the robustness that it will give the system. Once data is at the control node, it can easily be off loaded to a server. Using Bluetooth the user can theoretically go anywhere, and do anything (as long as the control node is in range) without fear of data being lost between the microcontroller and the control node.

2.2.3 WiFi

The use of a WiFi module at first appears to reduce the complexity of the project. However, the ability to get real-time data analysis and sync data across different WiFi access points presents a challenge requiring action by the subject being monitored. Given the goal of collectin data from Parkinson's Disease patients, the authentication of the WiFi module to a WiFi network presents an obstacle that cannot easily be overcome.

2.2.4 Zigby

Zigby is another protocol considered for transmitting data off of the microcontroller. It has been a challenger to Bluetooth and has found a niche in home automation. The primary challenge with Zigby is the lack of support from existing devices. Its primary benefit is the ability for it to form a mesh network.

2.3 Control Node

2.3.1 Overview

The control node acts as an intermediate device between the microcrontroller and webserver. It will regularly receive data from the microcontroller throughout the day and upload and process the data collected.

2.3.2 Smart Phone

The Smart Phone will the be intermediate device between the microcontroller and the server that collects the data. The smartphone will be connected to by the microcontroller with Bluetooth. Bluetooth is being used over WiFi because when WiFi is used there is a complication with connecting the microcontroller to the WiFi, while Bluetooth is easily used with any smartphone. Additionally, a follow on project to this project could be more processing at the control node level. A smart phone, could process the data itself, and immediately return it to the user for real time information. Lastly, one consideration is how convenient things are for the user. On a smart phone, it would be easier to integrate additional sensors with a user interface already existing. A goal is to make things hassle free to the end user, most people already carry a smart phone on them, adding an additional piece of hardware, such as a Raspberry Pi would just create more redundant hardware.

2.3.3 Raspberry Pi

A Raspberry Pi was considered as the intermediate device. A Raspberry Pi is a single-board computer, and therefore could easily act as the intermediate device. However, it would require a great additional support in order to get Bluetooth integrated into it, where with a smart phone, it is readily available. Additionally, the smart phone one day may be able to present the data real time data to the user.

2.4 Web Server

Many proven web technologies and frameworks exist for developing the web server. Options range from Python with Flask to Nodejs with Express. Each of the these web frameworks, Flask or Express, are highly extensible.

2.5 Database

The database for the project will likely be postgres. In addition to being a relational database it offers sufficient nosql options through a key value datastore.

2.6 Analysis Visualizations and Tools

Numerous tools exist for visualizing data through a website. For example, d3.js is a JavaScript library for making many forms of charts and graphs. Additionally, JavaScript libraries such as Crossfilter.js allow for client side multidimensional filtering and analysis.

3 Technology Integration

Integration between the different layers of the project is likely the most challenging apect of the project due to the need for custom protocols and the inherent bottlenecks. For example, creating a reliable, asynchronous application layer protocol on top of Bluetooth for synchronizing the microcontroller and node or creating a REST API for the node to communicate with the server.

The necessary protocols for integration are on the application layer and will make use of well-defined protocols such as HTTP, TCP and Bluetooth.

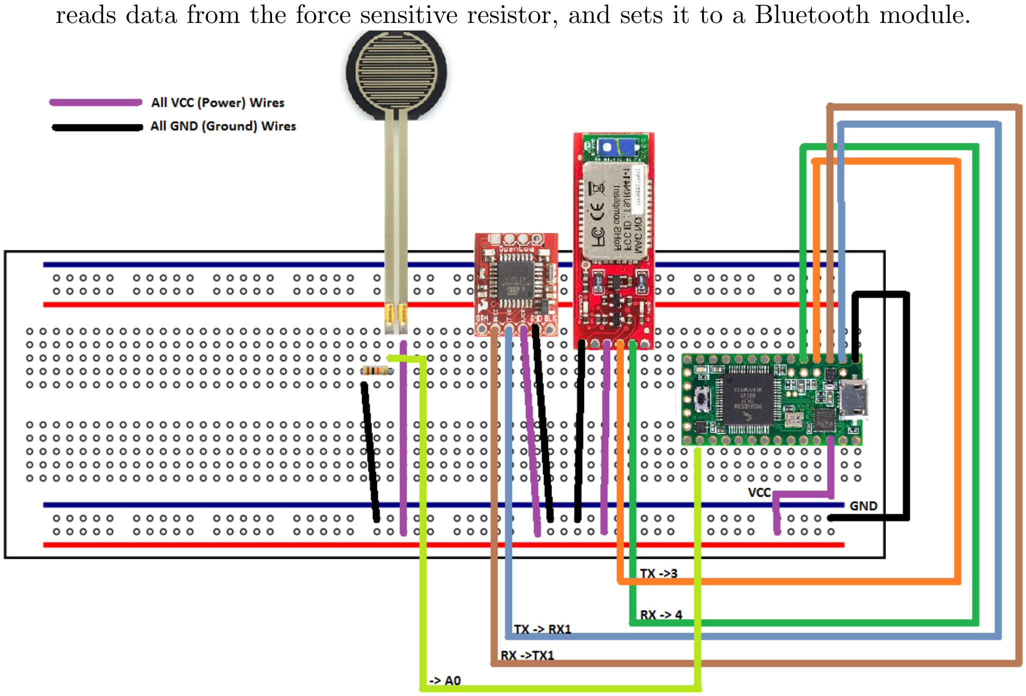
4 Proof of Feasibility

The proof of feasibility for this system currently exists in a few different pieces, that are currently working. There is a break between the control node and the database due to the lack of a proper phone application at this time.

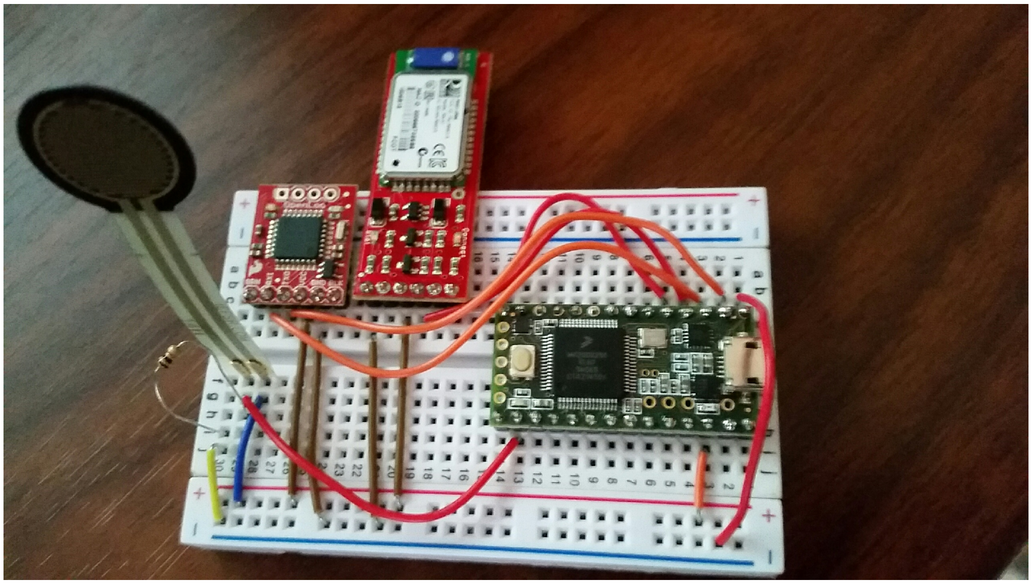
4.1 Microcontroller

Currently, all the sensors work as expected through the microcontroller. We are able to connect various sensors to the Teensy and collect data from each one of them. The microcontroller is also able to store that data in a buffer (SD card) in just a normal text file. Lastly, the microcontoller is able to push that data to a Bluetooth device and transmit the data from the buffer.

From here, their are only two pieces that need to be done at this level. The first is integrating more sensors into the system, and then placing them all it into a shoe as a wearable device. The main difficulty with this step will be making a device that is robust, and can withstand the both the weight of a person, and the human movements. The last piece, is programming a method that detects when the device is plugged in for charging. Once the device is plugged in, it will send all the data through the Bluetooth to the node. Concurrently with that, a method that will send all data in the event the buffer fills.
The image below is a detailed diagram on how to set up a working device that reads data from the force sensitive resistor, and sets it to a Bluetooth module.



The image below is the above diagram in real life. This device currently will read in the amount of force being applied to the force sensitive resistor, and write it to an SD card. The data being read and written is simply integers.

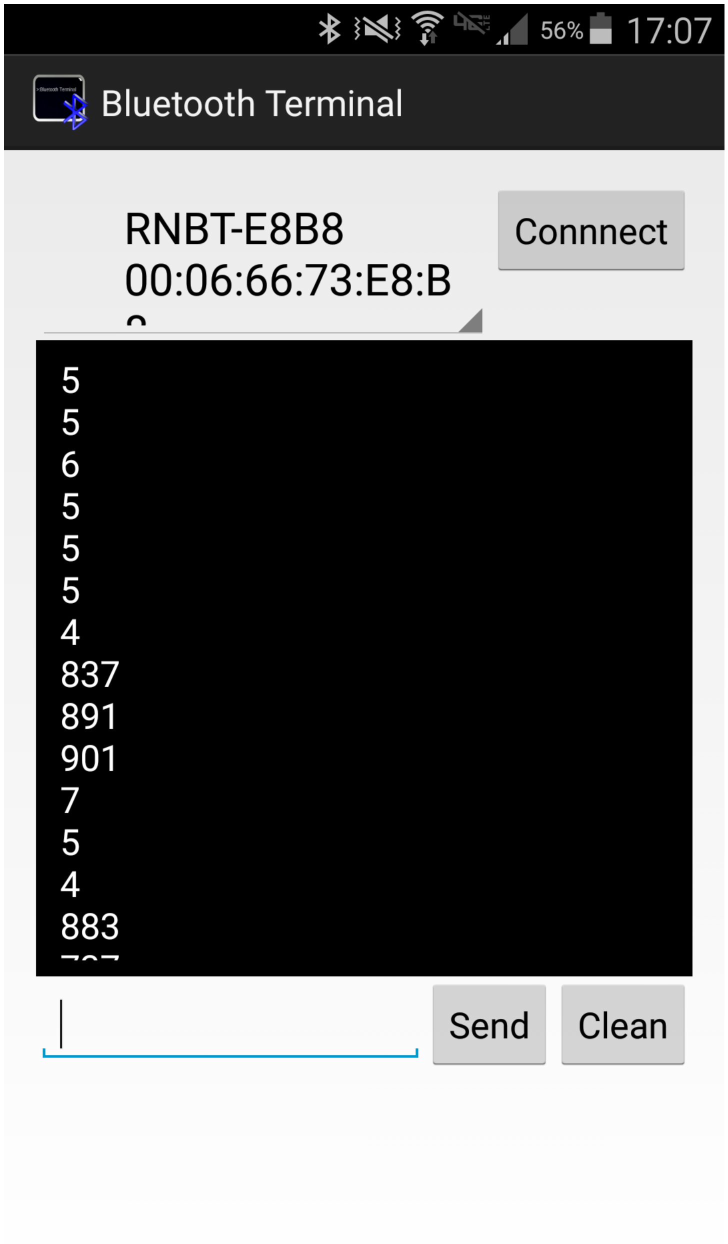


4.2 Microcontroller Data Transfer and Control Node

Using the microcontroller, the device is able to send data to a Bluetooth module on a phone uninterrupted, in a continuous stream of data. All this step requires, is opening any Bluetooth terminal application on a mobile phone. Once its open, pair with the device, and then the stream of data will begin, printing each piece of data in the terminal.

From here, the only one piece that is missing is the data needs to be saved on the phone, and sent to the web server. This requires us to write our own application, as there are no applications that will save that data, and connect to our specific server.

As seen in the image below, the Bluetooth module is connected to the Bluetooth terminal application. Once its connected, it immediately starts streaming data from a force sensor resistor. The force sensor resistor measures pressure on it. In the data shown below, the numbers in the 800 range are when a high pressure (finger pinch) is applied to the sensor.



4.3 Database

As sample data set containing 400 million rows was created in Postgres using a single table containing possible sensor values. The test values were created to form cycles with alternating peaks in two sensor values to simulate a gait. The window functions of Postgres were then used to calculate the sensor change and when the sensor delta changes direction approximating the gait cycle.