# Design Document, $V1.0$

Justin Poehnelt
Ahmad M. Meer
David Nagel
Jack Burrell

February 5, 2016

# Contents

# 1 Introduction

As information technology becomes more ubiquitous in our lives, researchers hope to find new and more complete sources of data. The question for the researcher quickly goes from how do I collect this data to how do I collect, manage and analyze an extremely large data set.

This project hopes to answer these question by providing a framework for the current and future research needs of the project sponsor, Dr. Kyle Winfree. His research is focused on the analysis of human movement, specifically the issues associated with Parkinson's Disease.

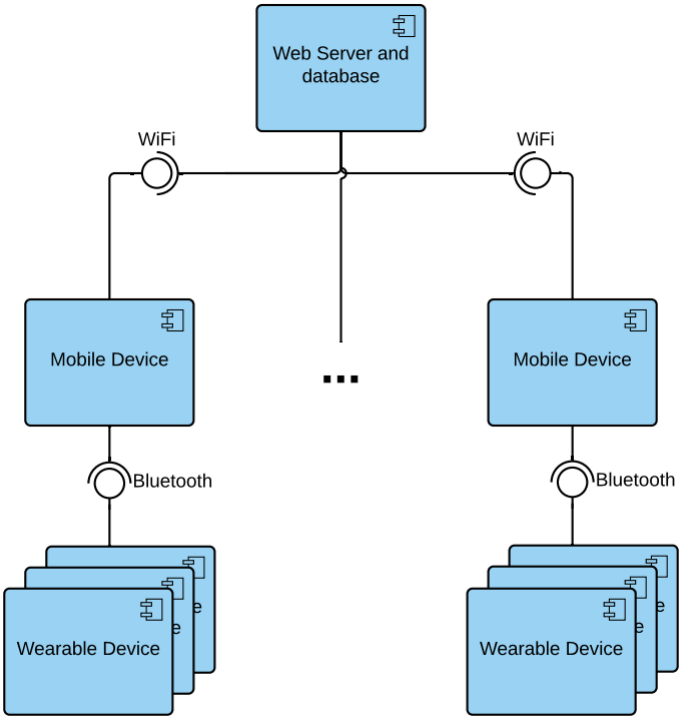
While many devices are capable of tracking the count of steps an individual takes, this project aims to provide a framework for collecting data at a much higher frequency with hundreds of samples collected every time a subject takes a step. This requirement places a significant burden on the hardware and software of the layered system that has been proposed. The sheer amount of data presents a great challenge to the team.

The most resource constrained layer of the system is the microcontroller and associated sensors placed on the subject, hereafter referred to as the wearable device. The wearable device must be able to perform a number of functions at full-capacity and any inaccurate or poorly synchronized data will lead to failure. Fortunately the risks identified in the other layers are more easily mitigated.

This document gives a more in depth look into the overall architecture of the entire system. It will also take a more detailed look into the architectures of the wearable device, and the mobile device. It will also provide a complete overview of how each component interacts with each other, and the larger system.

## 2 Architectural Overview

In order to meet these challenges, our system will consist of a group of smaller systems. The overall architecture pattern of this system is a repository. At the top, our repository is a web server and database that collects all the data. Here, the data can be accessed for general use by a user. Branching from the repository, there are a number of identical subsystems that gather this data.



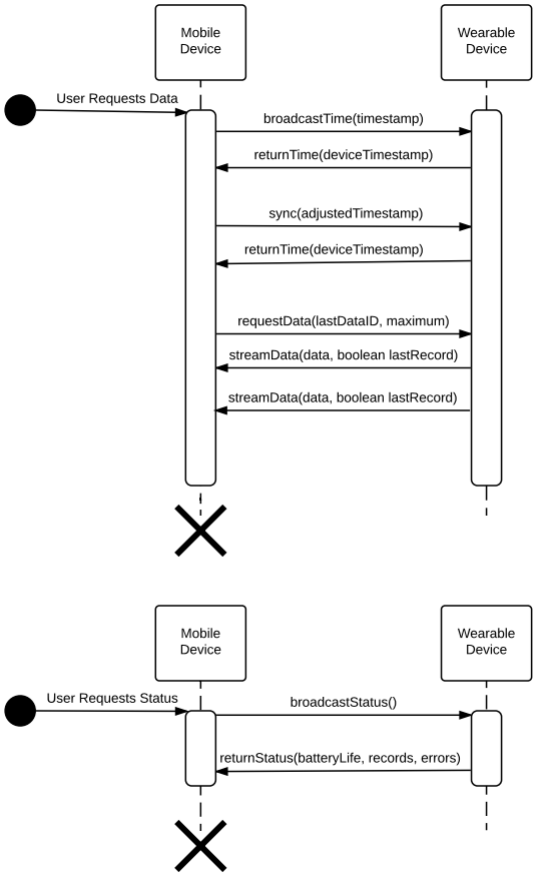

Each subsystem consists of two parts, the mobile device and the wearable device. While each wearable device can have a verity of different sensors, it will interact with the mobile device the same way regardless. The mobile device acts as a intermediate point between the wearable device and the web server. There are a number of reasons as to why this intermediate point exists.

1. The mobile device provides the end user with a limited amount of control over the wearable device. The user cannot interact with the wearable device directly.

2. The mobile device acts as a secondary storage point for large amounts of data that can no longer be stored on the wearable device.

3. in order to give the end user the ability to move wherever they want, it is much simpler to send data from the mobile device to the web server over WiFi. Connecting the wearable device directly to a WiFi connection would require preprogrammed WiFi addresses.

The most advance piece of communication of the system is between the wearable device, and mobile device over a bluetooth connection. There are two instances in which this communication takes place. Either the mobile device requests data, or the mobile device requests the status of the wearable device. In order to keep data organized in a logical order, and in order to have it be able to be synced between wearable devices, it must have a timestamp. That timestamp comes from a handshake between the two devices.



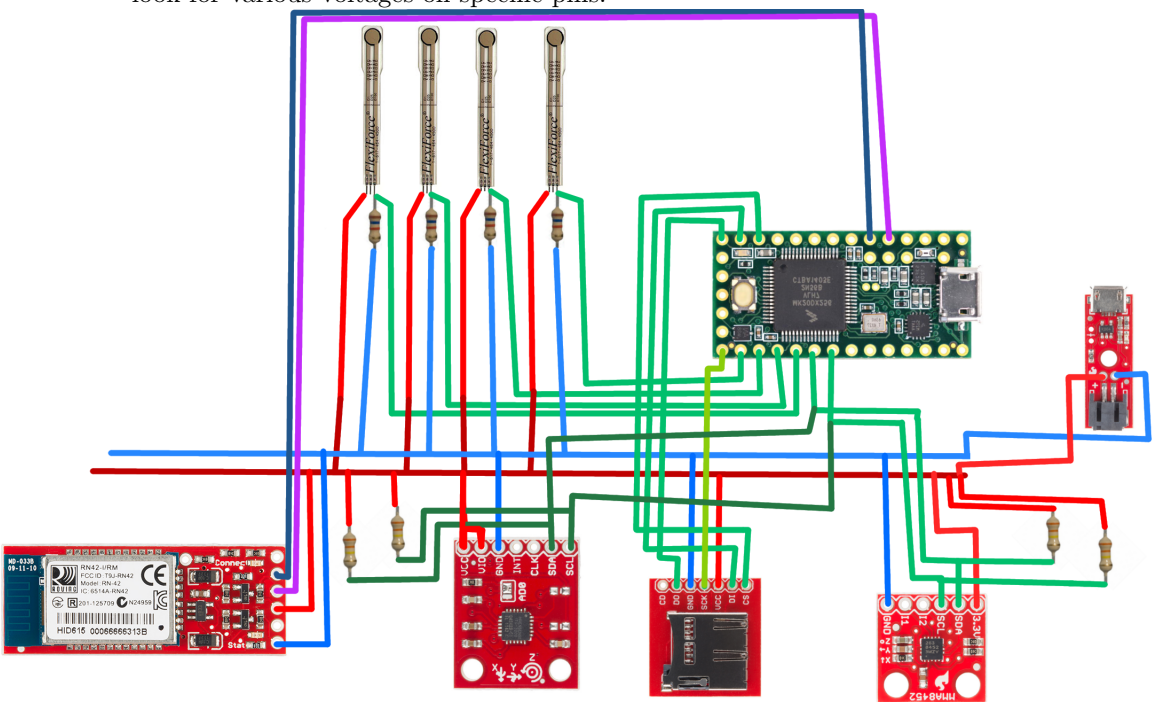

The top figure shows the handshake process between the two devices. Once they are synced with each other, data is sent continuously until the wearable device sends "lastRecord" as true, when the last record is being sent.

The communication for the status is much simpler. When the wearable device receives that request, it will see that it is a status request, and simply return return the proper data without question. It does not expect any response in return.

# 3 Module and Interface Descriptions

## 3.1 Wearable Device

First and forest most, one of the most difficult aspects of the wearable device is the design of the hardware itself. The below diagram shows how sensors connect to the microcontroller. It is very important that these sensors are hooked up this way. Many times, the software on the microcontroller will specifically look for various voltages on specific pins.

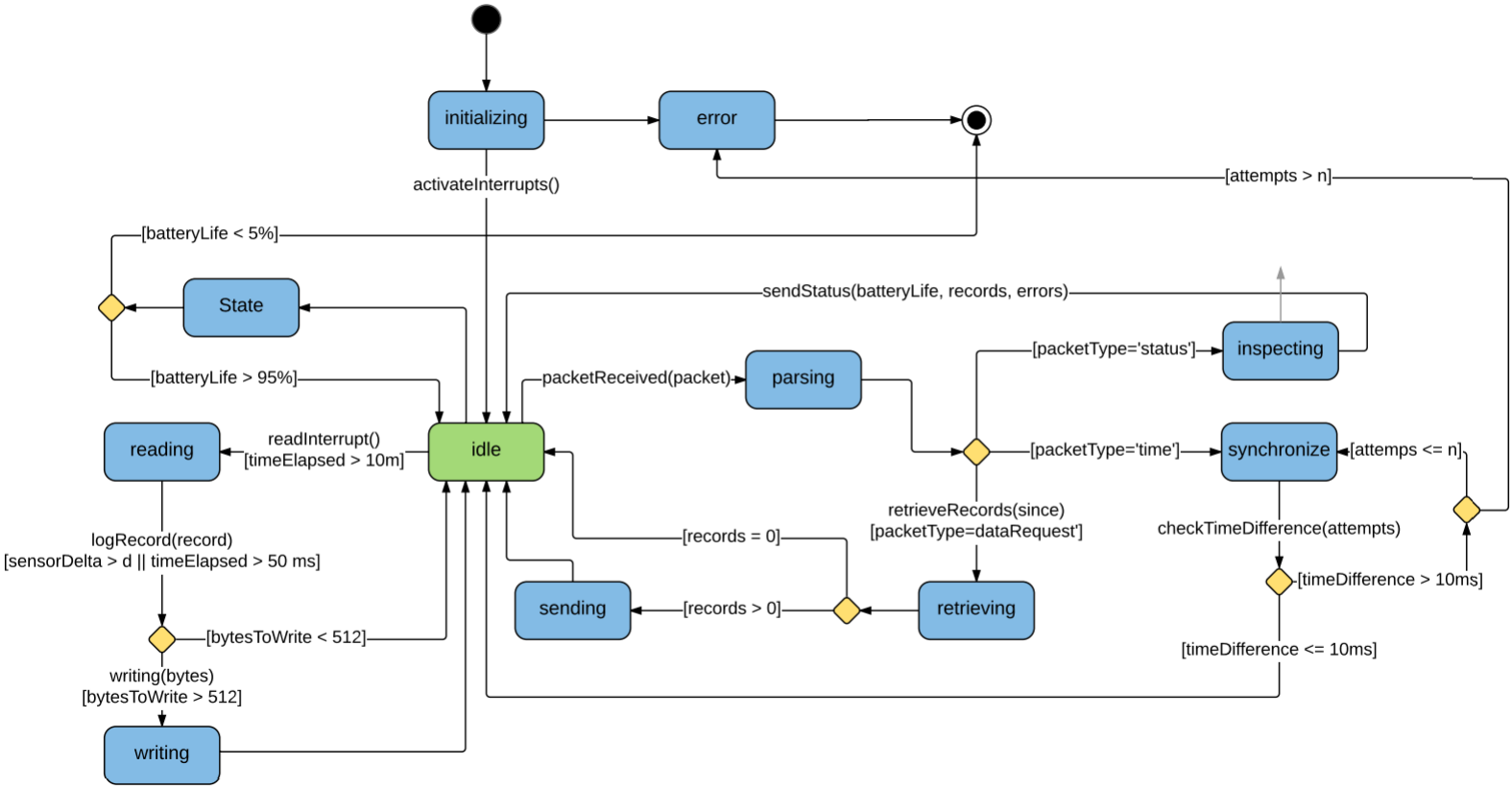


The wearable device uses a three-tier architecture style. This architecture style separates its functions into separate layers in which the user has different level of interaction with at each level. The first layer, the bottom layer, is the data tier, or hardware layer. Here is where data is collected and stored. This exists in the form of force sensitive resistors, gyroscopes, accelerometers, a bluetooth module and an SD card that collects the data, and stores it locally. Example data gathered from sensors can look like the following figure.

| ms | gyro x | gyro y | gyro z | accel x | accel y | accel z | fsr1 | fsr2 | fsr3 | fsr4 |
|---|---|---|---|---|---|---|---|---|---|---|
| 13001 | -255 | 0 | 247 | 0.099 | 1.299 | 0.499 | 0 | 8 | 0 | 0 |
| 13014 | -255 | 0 | 247 | 0.099 | 1.299 | 0.499 | 0 | 9 | 0 | 0 |
| 13027 | -255 | 0 | 247 | -0.166 | 1.034 | 0.234 | 0 | 9 | 0 | 0 |
| 13040 | -255 | 0 | 247 | -0.124 | 1.076 | 0.276 | 0 | 9 | 0 | 0 |
| 13053 | -255 | 0 | 247 | -0.113 | 1.087 | 0.287 | 0 | 9 | 0 | 0 |
| 13066 | -254 | 0 | 247 | -0.165 | 1.035 | 0.235 | 0 | 10 | 0 | 0 |
| 13079 | -254 | 1 | 247 | -0.102 | 1.098 | 0.298 | 0 | 14 | 0 | 1 |
| 13092 | -253 | 1 | 247 | -0.434 | 0.766 | -0.034 | 0 | 14 | 0 | 4 |
| 13105 | -253 | 1 | 247 | -0.434 | 0.766 | -0.034 | 0 | 14 | 0 | 5 |
| 13118 | -255 | 1 | 254 | -0.434 | 0.766 | -0.034 | 0 | 14 | 0 | 5 |
| 13131 | -255 | 1 | 254 | -0.834 | 0.366 | -0.434 | 0 | 14 | 0 | 6 |
| 13144 | -254 | 2 | 254 | -0.834 | 0.366 | -0.434 | 0 | 14 | 0 | 5 |
| 13157 | -254 | 0 | 254 | -0.634 | 0.566 | -0.234 | 0 | 14 | 0 | 5 |
| 13170 | -252 | 0 | 254 | -0.675 | 0.525 | -0.275 | 0 | 13 | 0 | 6 |
| 13183 | -253 | 0 | 254 | -0.545 | 0.655 | -0.145 | 0 | 4 | 0 | 1 |
| 13196 | -255 | 0 | 254 | -0.525 | 0.675 | -0.125 | 0 | 0 | 0 | 1 |
| 13209 | -255 | 1 | 254 | -0.563 | 0.637 | -0.163 | 0 | 0 | 0 | 1 |
| 13222 | -255 | 1 | 254 | -0.261 | 0.939 | 0.139 | 0 | 0 | 0 | 0 |
| 13235 | -255 | 1 | 254 | -0.266 | 0.934 | 0.134 | 0 | 0 | 0 | 0 |

The second layer, or the logic tier, is the section that handles how the device actually operates. Since this is an embedded device, it primarily revolves around a deamon, that continuously runs. This is because of the granularity of the data required. Since the data clarity needs to be so high, this loop will never stop, it will continuously collect the results of the hardware level, structure it, and then store it locally. The only way this daemon will stop is if the system detects that there is no reason to collect data (user is sitting, or user is clearly not wearing device), in which it will go idle, or there is a specific request for data or a status from the mobile device.

The top level, is the presentation layer. One of the goals of this project, is to have the user interact with the device as little as possible, due to the type of end user that will use this system. However, there are some ways in which the user can interact with device. This mainly comes from signals sent from the mobile device. These signals will mostly be data requests, but can expand to more controls in the future. Lastly, this level is where the wearable device performs a handshake with the mobile device when sending data. The wearable device is synced with the mobile device at this high level. Once the two are synced, the daemon will shift from collecting data, to sending data.



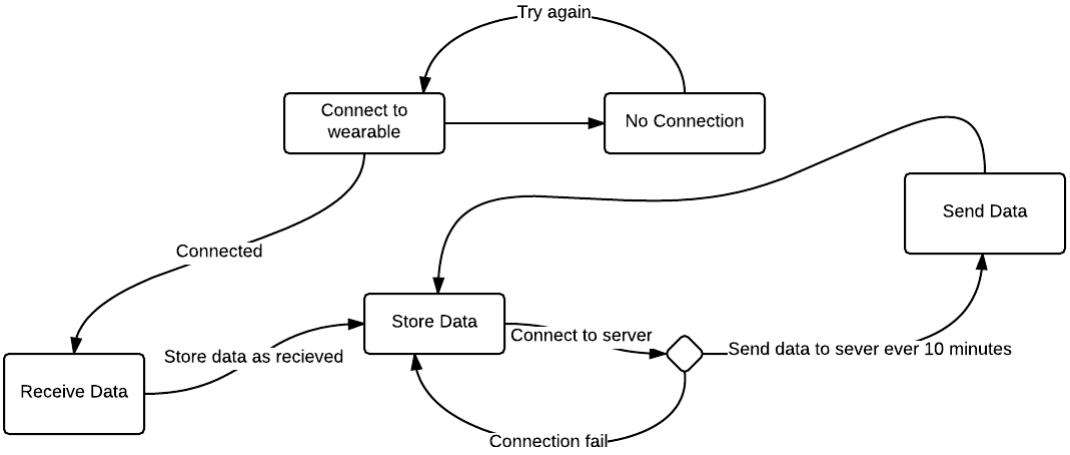

## 3.2 Mobile Device

The mobile device uses the model-view-controller architectural style. This architecture style separates its functions into three separate components in which the user only interacts with one. The model captures the behavior of the application. The model is the where the data received from the wearable device is sent to and stored. This will exist as a database table on the mobile application.

The view is the output of information displayed to the user. This is where the user will be able to see the data being sent from the wearable device to the mobile application in real time, and a table populated with data collected. The user will also be able to see a visual representation of the data in some different oriented graphs.

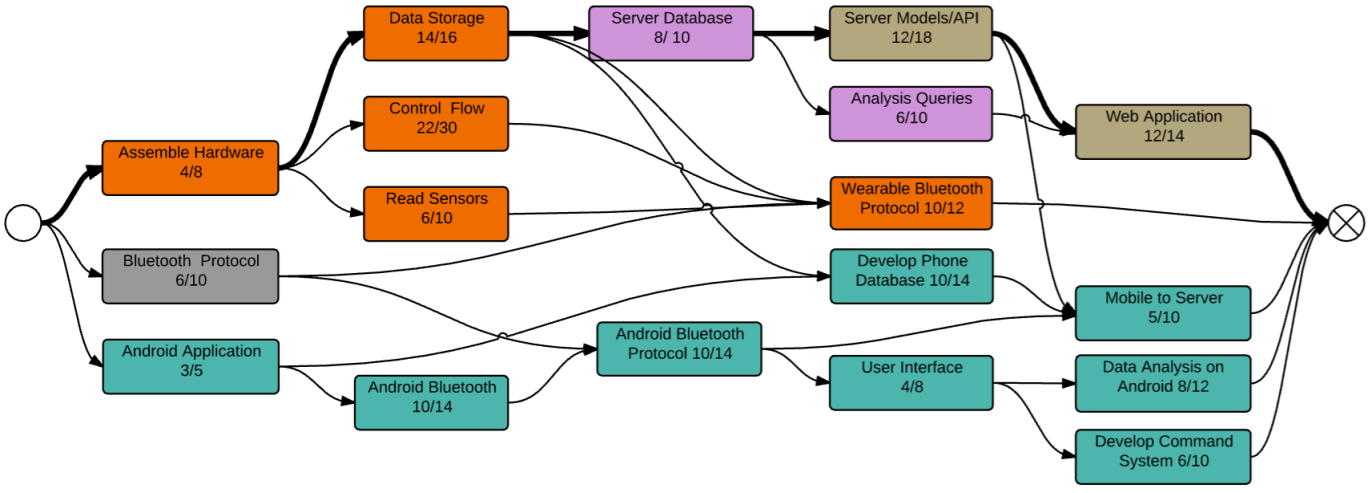
The controller accepts inputs from the user and converts them into commands for the view and model. The controller will allow the user to be able to connect

to the wearable device. The user will also have access to multiple different options on how to view the data.




# 4 Implementation Plan

For implementation, the team devised a PERT chart in order to determine what components rely on each other. This determined the overall flow of the project. Ultimately, the mobile application and wearable device can be started separately (preferably by two persons for each) as they exist independently until closer to the end.



Additionally, an analysis of the time required for each component was done. This was done by an estimation of the earliest a component could be completed, and the latest. This also provided the benefit of providing slack time, or time that a component could be delayed, without affecting the whole project. The total estimated time for the best case scenario is that the project will take 58 days to complete.

| | | Start Day | | |
|---|---|---|---|---|
| | **Days** | **Earliest** | **Latest** | **Slack** |
| Wearable Device | | | | |
| Assemble Hardware | 6 | 0 | 0 | 0 |
| Data Storage | 15 | 6 | 6 | 0 |
| Control Flow | 26 | 6 | 21 | 15 |
| Read Sensors | 8 | 6 | 39 | 33 |
| Bluetooth Protocol | 11 | 32 | 47 | 15 |
| Design | | | | |
| Bluetooth Protocol | 7 | 0 | 23 | 23 |
| Mobile Device | | | | |
| Android Application | 4 | 0 | 14 | 14 |
| Android Bluetooth | 12 | 4 | 18 | 14 |
| Android Bluetooth Protocol | 12 | 16 | 30 | 14 |
| Mobile Database | 12 | 21 | 38 | 17 |
| Mobile to Server | 8 | 45 | 50 | 5 |
| User Interface | 6 | 28 | 42 | 14 |
| Data Analysis on Mobile | 10 | 34 | 48 | 14 |
| Command System on Mobile | 8 | 34 | 50 | 16 |
| Server and Database | | | | |
| Server Database | 9 | 21 | 21 | 0 |
| Analysis Queries | 8 | 30 | 37 | 7 |
| Server Models/API | 15 | 30 | 30 | 0 |
| Web Application | 13 | 45 | 45 | 0 |