**91.411.201: Software Engineering I**

**Team #1**

**Firmware**

**Software Requirements Specification**

**Document**

**Version: 1** **Date: (04/15/2014)**

**Table of Contents**

**1. Introduction**

*1.1 Purpose*

*1.2 Scope*

*1.3 Definitions, Acronyms, and Abbreviations*

*1.4 References*

*1.5 Overview*

**2. The Overall Description**

*2.1 Product Perspective*

2.1.1 System Interfaces

2.1.2 Interfaces

2.1.3 Hardware Interfaces

2.1.4 Software Interfaces

2.1.5 Communications Interfaces

2.1.6 Memory Constraints

2.1.7 Operations

2.1.8 Site Adaptation Requirements

*2.2 Product Functions*

*2.3 User Characteristics*

*2.4 Constraints*

*2.5 Assumptions and Dependencies*

*2.6 Apportioning of Requirements*

**3. Specific Requirements**

*3.1 External interfaces*

*3.2 Functions*

*3.3 Performance Requirements*

*3.4 Logical Database Requirements*

*3.5 Design Constraints*

3.5.1 Standards Compliance

*3.6 Software System Attributes*

3.6.1 Reliability

3.6.2 Availability

3.6.3 Security

3.6.4 Maintainability

3.6.5 Portability

*3.7 Organizing the Specific Requirements*

3.7.1 System Mode

3.7.2 User Class

3.7.3 Objects

3.7.4 Feature

3.7.5 Stimulus

3.7.6 Response

3.7.7 Functional Hierarchy

*3.8 Additional Comments*

**4. Change Management Process**

**5. Document Approvals**

**6. Supporting Information**

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to give a description for the requirements for the hardware/firmware part of the cloud based fall detection project. This document will explain the constraints of this subsystem and what interactions it must have with other external systems. The intended audience of this document are the developers of the system who will ultimately implement all of its parts.

## 1.2 Scope

The “Cloud Fall Detection system” is a system that uses a body worn device with sensors to keep track of a person's health and make their health history available to doctors and other personnel anywhere they are using cloud technology.

People rehabilitating at home can wear the device and their doctor can make sure that the rehabilitation is going well without actually having to visit the patient. The doctor will get a notification if there are any problems with rehabilitation so that the patient can stay at home safely while rehabilitating.

Because the doctor can see all of the patient’s history on the cloud he can recommend changes to the patient’s diet or any other changes to help with the rehabilitation as he will be able to see patterns on the patient’s rehabilitation history record. If the patient falls the Doctor or hospital will be immediately notified.

## 1.3 Definitions, Acronyms, and Abbreviations.

* **Cloud:** Servers on the internet processing and storing data we send them.
* **Bluetooth:** short range wireless communication.
* **Wireless:** Data communication over the air using radio frequencies.
* **Microcontroller:** a mini-computer that has everything (from ram to cpu) integrated on a single chip.
* **Android:** Operating system run by some cellphones.
* **Accelerometer:** Sensor that measures acceleration.
* **Gyroscope:** sensor that measures orientation.
* **TCP/IP:** communication protocol
* **Server:** Computer which serves client requests
* **JSON:** a format for transmitting data

## 1.4 References

Title: Mobile accelerometers and gyroscopes explained

Date: 4/12/2014

Organization: O'Reilly answers

Source:[*http://answers.oreilly.com/topic/1751-mobile-accelerometers-and-gyroscopes-explained/*](http://answers.oreilly.com/topic/1751-mobile-accelerometers-and-gyroscopes-explained/)

Title: Cloud computing

Date: 4/12/2014

Organization: Wikipedia

Source:[*http://en.wikipedia.org/wiki/Cloud\_computing*](http://en.wikipedia.org/wiki/Cloud_computing)

Title: *Microcontroller*

Date: 4/12/2014

Organization: Wikipedia

Source: *http://en.wikipedia.org/wiki/Microcontroller*

*.*

## 1.5 Overview

The rest of this document contains 5 parts 2 of which are the most important and those are section 2 and section 3. Section 2 provides a high level overview of the system and how it interacts with other systems, it also points out the constraints of the system. Chapter 3 is the actual requirements specifications for developers to take into account when developing the system. It describes the interfaces to the system and the functional and nonfunctional requirements.

# 2. The Overall Description

This section gives an overview of the entire system. We will explain how the system interacts with other systems and how it works and what it does to accomplish its tasks.

There is really just one type of user for this sub-system which is the patient but since this is an embedded system there is minimal user interaction so we will focus on the other aspects of the system. We will also make explicit any assumptions we are making such as the user using an android cellphone etc...

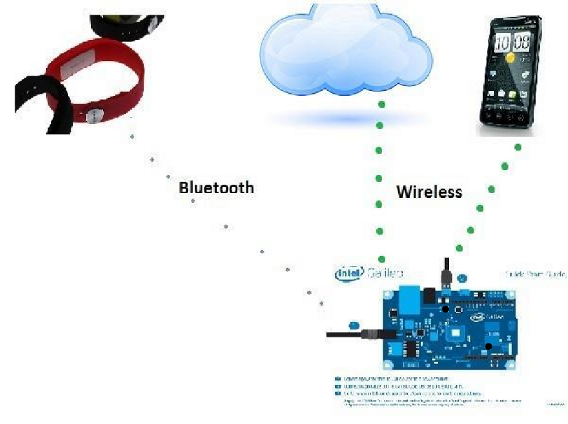
## 2.1 Product Perspective

The system as a whole has many different parts: cloud computing, microcontroller, wristwatch with sensors and a mobile app. Our sub-system is an embedded system which is at the beginning of the whole. Our sub-system is a SOC/microcontroller which has wireless and bluetooth capabilities; it uses bluetooth to read the sensor values from the wristwatch the person wears and does some pre-processing on that data before sending it to the cell phone application.

The cellphone application displays the data in a meaningful manner and performs fall detection on the data and if a fall is detected it connects to the cloud and notifies the doctor.

If time permits we will also do some basic fall detection on the microcontroller and bypass the cellphone communication if a fall is detected.

Our Intel Galileo board/microcontroller and the external devices/systems it interacts with:



### 2.1.1 System Interfaces

The following is a list of the interfaces to the system.

### 

### 2.1.2 Interfaces

There really is no user interface for this sub-system since it is an embedded system and requires minimal or no user interaction after it has been correctly set-up.

### 2.1.3 Hardware Interfaces

There are two hardware devices used for this system: a wristwatch which a person can wear and that has sensors such as an accelerometer and gyroscope in it so that the values of the sensors can be used to determine if the user has fallen on the floor. The wristwatch broadcasts the values over bluetooth so that they can be easily read for processing.

The second piece of hardware is a SOC/microcontroller that has bluetooth and wireless capabilities so that it can connect to the wristwatch and read the sensor values and and do some pre-processing in them and detect if a fall occurs and then send the data to the patient’s cell phone for further processing. If the microcontroller detects a fall it will bypass the cellphone and directly connect to the servers on the cloud to notify the doctor taking care of the patient.

The microcontroller needs to be on the same network and the cellphone and the wristwatch needs to be paired with the microcontroller so that all communications can work smoothly. As of now only open source wearable sensor watches and android phones will be supported, and the communication between the wristwatch and the microcontroller will use RFCOMM when possible and the communication with the cell phone will use TCP.

### 

### 2.1.4 Software Interfaces

**Name: python-bluetooth**

Mnemonic: pybluez

Version number: 0.18

Source: <https://code.google.com/p/pybluez/>

Purpose: To make it easier to use the bluetooth adapter through python since this library created python wrappers around system Bluetooth resources.

Message content and format: the message content will be accelerometer and Gyro values from the wristwatch and the format is specified by the format the wristwatch uses.

### 2.1.5 Communications Interfaces

For communication we will use Bluetooth to communicate with the wristwatch and wireless to communicate with the patient’s cellphone. The wristwatch has sensors in it that monitors the person's movements and it broadcast’s the sensor values over bluetooth. Our microcontroller will use bluetooth to receive those accelerometer values due some preprocessing on the values and hand it over to the patient’s cell phone over TCP for further processing. If a fall is detected the system will automatically connect to the server in the cloud and report it.

### 2.1.6 Memory Constraints

128 MB should be enough for this sub-system. Since we are just processing accelerometer and gyro values we don't need much memory to preprocess and forward it, but we might need to store a small buffer or database of values that cannot be processed right away.

### 2.1.7 Operations

Usually all the user has to do is to make sure that their cellphone is on the same network as the microcontroller and that their wristwatch is paired which the microcontroller, After that no user interaction is required. the microcontroller will process the accelerometer and gyroscope data as the user moves and if a fall is detected it will be reported otherwise the data is forwarded to the user’s cellphone.

### 2.1.8 Site Adaptation Requirements

No site setup is required of a user beyond an internet connection.

## 2.2 Product Functions

The system, broken up into three major components, performs, for the most part, one job per component.

1. The sensors, worn as a watch on the patient’s wrist, collects data and sends this raw data to a microcontroller.
2. The microcontroller will process the data it receives and send it to the patient’s smartphone OR directly to the cloud, depending on whether critical information has been detected, such as a patient fall.
3. The smartphone will store the data for future use by, for example, the patient’s doctor. How the data is retrieved, displayed, etc. is up to the individual implementation of the app.

## 2.3 User Characteristics

The usual user of this system is any type of person who is doing home rehabilitation. They should be familiar with how to use a cellphone and that is all that is needed since there is no real user interface.

## 2.4 Constraints

The following are the restraints we must abide by:

* FDA regulates medical devices
* Real time processing of data
* Must interface with cloud and patient’s cellphone
* Fall detection must be accurate
* Critical that the system collect all data
* Data must be securely transmitted

## 2.5 Assumptions and Dependencies

**Assumptions about the system:**

* User has an Android cell phone
* Microcontroller is Linux based
* Sensor data broadcast over bluetooth

## 2.6 Apportioning of Requirements.

Some of the features for new versions or in case the project is late are:

* Communication with cloud servers
* Fall detection

# 3. Specific Requirements

## 3.1 External Interfaces

The system by itself requires as little input as possible, so external interfaces are minimal. The system shall only send external data via the cloud by the patient’s smartphone over TCP, so this shall be the only method from which external systems can expect to receive data. This information shall be processed data related to the patient, such as whether or not they have fallen or how many steps they have taken.

## 3.2 Functions

* System shall read Gyro and Accelerometer values from wearable sensor wristwatch
* System shall forward preprocess data to patient’s cell phone over TCP
* System shall detect if a fall occurred and if so communicate with cloud server
* system shall ensure connections to cell phone and cloud server
* system shall be able to handle connection loss

## 3.3 Performance Requirements

At the moment only one user will be supported but the code must allow for the number of users to grow, we will process accelerometer and gyroscope data and how much is processed is only limited by the link speed, 99% of transactions shall be processed in less than 1 second,

## 3.4 Logical Database Requirements

N/A

## 3.5 Design Constraints

We will only be able to provide data that we are able to get from the sensors. To this end we obviously cannot check patient statuses or conditions that cannot be determined by a combination of the sensors provided. We must also keep in mind the limited resources available, as this is an embedded system.

### 3.5.1 Standards Compliance

The existing standards we are required to comply to are only any existing Bluetooth and TCP standards, as they are what we will be using for communication.

## 3.6 Software System Attributes

### 3.6.1 Reliability

The system requires little setup, so it will be reliable almost 100% of the time. The only setup required would be to establish a connection between the sensors and the microcontroller, and the microcontroller and the patient’s smartphone. After that nothing else is required of the user.

### 3.6.2 Availability

The device will run constantly, as it monitors patient information. There may be watchdog-like systems to restart any crashed processes if necessary, but beyond that, it would be easy for the user to simply restart the device. However, we do not plan on major system failures, as both the sensors and the microcontroller run very simple code that has little risk for catastrophic system failure.

### 3.6.3 Security

As communication with the devices will be with either Bluetooth or TCP, any data is inherently secure and safe from malicious access. And even then, there is very little sensitive data being sent around--it’s all simple sensor information and interpreted sensor information. Nothing specific to the patient, other than what the sensors think the patient is doing, is passed from device to device.

### 3.6.4 Maintainability

Required maintenance should be minimal, as this is an embedded system, and it is thus very difficult, if not impossible, to provide software updates without being intrusive or requiring that the user replace the entire device. At most we can provide an interface to the smartphone that is designed for the possibility of being forwards-compatible, but beyond that, there is little that is possible.

### 3.6.5 Portability

Portability will largely be limited to what we are able to achieve with the smartphone. Because we are using a specific device to gather data such as temperature or motion, any interfaces we design for that device are almost guaranteed to be non-portable. This is inherent in the design. However, as just mentioned, we may be able to achieve some degree of portability with the smartphone. Depending on how generic we can make our interface calls, it may be possible to simply send data without any idea of what will be receiving it. In this way, we would be able to send data to any kind of smartphone that can receive the type of data that we send.

* The sensors must, by design, have host-dependent code. This is a significant portion of the project.
* The choice of language is Python, which is fairly portable.
* As Python portability is dependent on the interpreter, the only requirement is that there be such an interpreter for the system we are developing.
* We plan on using Linux as an operating system, which is very portable.

## 3.7 Organizing the Specific Requirements

### 3.7.1 System Mode

Only one mode of operation.

### 3.7.2 User Class

Only one type of user: rehabilitating patient.

### 3.7.3 Objects

Only object we will use is a JSON object to represent the Gyro and Accelerometer values of the wristwatch which then can be sent to the cellphone for unpacking and processing.

### 3.7.4 Feature

Person falls - message is sent to the cloud

### 3.7.5 Stimulus

The system is running constantly, so no stimulus is required.

### 3. 7.6 Response

The only response in our system happens when a fall is detected, we then connect to the cloud server and notify it of the event.

### 3.7.7 Functional Hierarchy

No functional hierarchy diagrams are necessary as no components inherit from each other. The sensors, microcontroller, and smartphone are largely independent from one another and, besides communication, share little in common.

## 3.8 Additional Comments

None

# Change Management Process

Changes to the SRS are made after the team has reached a consensus and we all agree that the changes are reasonable and deliverable in the required amount of time. Since we are a small team the changes will be submitted by the customer by email phone or any other communication medium and are received and reviewed by us directly.

# Document Approvals

This document has not yet been approved.

# Supporting Information

Supporting information can be found by consulting the relevant standards and technologies that this product uses.

* Information related to Python can be found at https://www.python.org
* Information related to TCP can be found at the RFC Index
* Information related to Bluetooth can be found at <https://www.bluetooth.org>
* Information related to the pybluez library can be found at its Google Code repository
* Information related to Android development can be found at developer.android.com