HALMSTAD UNIVERSITY

Computer Engineer, 180 credits

DEGREE THESIS

Cloud based platform for real time Gait analysis

System component: Mobile architecture

Tim Svensson

Computer science and engineering, 15 credits

Halmstad 2017-06-20

Abstract

Today gait analysis are performed in laboratories with expensive equipment and people must visit the labs to perform several supervised tests. The goal of this project is to develop a platform enables gait analysis with the accelerometer sensor in a mobile phone. This would allow more people to do gait analysis, as smart phones are widely available and cheap equipment compared to lab equipments. In order to solve this task a mobile application and a cloud server was created. The mobile application can gather data from the internal accelerometer sensor and a medical grade sensor simultaneously and send the data to the cloud server. When two sensors are used the symmetry between the left foot and the right foot can be measured, although the system works with only one sensor aswell.

On the cloud server the accelerometer data is analysed and gait analysis is done on the data and visualized on a web page. The mobile application can collect data for 4 hours at a sampling rate of 120Hz and two sensors are used. When sending data collected from two sensors to the cloud at a sampling rate of 120 Hz the amount of data is approximately 21.96 Mb/h. The goal of the project was to create a proof-of-concept platform to do gait analysis and that goal was fulfilled and a fully functional platform was developed.

Contents

1 Introduction

The gait cycle is described as the time between successive foot contacts of the same limb. In one gait cycle there are 2 steps and 1 stride. A step is defined as the time of a heel strike of one foot and the heel strike of the other. A stride is completed after two successive heel strikes on the same foot. Two important events in a normal gait cycle are the heel strike and toe off. They are essential in estimating stride and step parameters and that is why being able to detect these events is of great importance in any gait analysis application[1].

Gait is normally centered around three main components: locomotion, balance and ability to adapt to the environment. In the human body this is achieved through a balance between various interacting neuronal and musculoskeletal systems[2]. If the function of these systems were to be compromised, effects of this will be apparent in gait analysis.

As of today, gait analysis is typically done in lab environments with expensive and large equipments such as pressure-sensor walkways, force-platforms and 3D motion capture[3]. Although the analysis that is done in lab environment provides rich and accurate information it is not suitable for use in daily life, they are immobile and requires competent personnel to supervise and the information that are received are for short sessions of walking[4].

Recent years has seen progress in the field of small and lightweight Micro-machined Electromechanical Systems(MEMS) which enabled the inertial sensors to be integrated into small devices and consume lesser amount of battery.[5]. With MEMS technology it made the sensors small, inexpensive, durable and highly mobile which enables long-term collecting of data. Researchers have developed gait event detection algorithms from accelerometer and gyroscope data[6].

With these new gait event algorithms a new market opens up to be able to do gait analysis on smaller and inexpensive devices, such as smartphones. Also this opens up the opportunity to do gait analysis in daily life, such as outdoor walking.

1.1 Purpose and goal

This part of the project is about solving the problem of collecting data from inertial sensor and from medical grade sensor to a smartphone over a long period of time and process the data to a cloud server. The cloud part of the project which includes gait analysis and server architecture is discussed in another report called; System component: - Server architecture.

There is demand for a framework for performing gait analysis in real-world environments. Some general purpose platforms for performing scientific experiments such as e-Science central have been proposed[7]. However, there remains a challenge regarding integrating applications that rely on a rich runtime environment, which is typically the case for real time gait analysis algorithms. One of the fundamental challenges in such a project entails getting these algorithms to run in a cloud environment so it can be deployed anywhere. Making a specialized platform for gait analysis could also allow for a more tailored architecture and targeted optimization.

Another challenge would be collecting data through the mobile application, as the collection would be taking place over long periods of time. Also, as the algorithm is compiling in the cloud, hence the data needs to be sent there for processing. This makes it a challenge because the connection to the Internet can vary from being good to worse or sometimes even no connection at all.

The goal of this project is to develop a proof of concept Gait Platform that can analyse data over long periods of time. The analysis is going to be ”real-time” and results shall be displayed on a web page. The mobile application is going to act both as the collector of data and as the gateway to the cloud server.

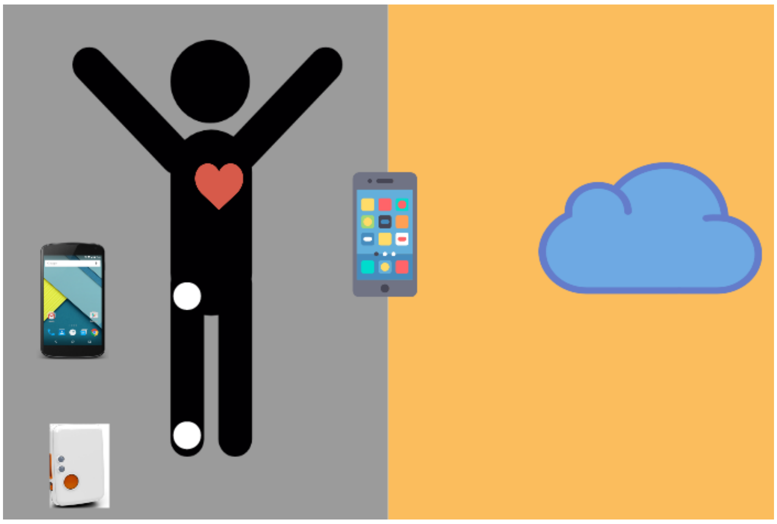


Figure 1: Gait Platform

Some examples how the platform can enable different kinds of gait analysis are

1.1.1 Example 1 - Rehab

If a person has an injury that makes them having difficulty in their walking they need to visit a doctor to ensure that the injury is healing and that their walking are getting better. The difficulty is that the doctor can only measure the walking when the patient is at the clinic. With the proposed platform the patient could mount a smartphone on his limb and start walking at home or outdoor. The doctor could then watch the result of the walking on his computer in near ”real-time”. This also enables to get results from the patience in between the visits to the clinic. The data the doctor analyzed can give suggestions on how the patience injury recovering is progressing.

1.1.2 Example 2 - Diseases

Parkinson’s disease affects the patient central nervous systems. The symptoms generally evolves slowly over time and at a early stage in the disease the patience normally get shaking, slowness in movement and difficulty with walking[8]. Studies has been made on Parkinson’s patients in an early stage of the disease before any visible disability has occurred. In these test it shows that PD patients had significantly different gait parameters comparing to healthy test objects. The test was done using wearable sensors and the result showed that gait can be analysed with more simple equipment yet in a effective way[9]. One possible outcome of this project could be to measure people who still don’t know they have PD to see if they are in danger zone of having the disease.

1.2 Research Questions

Implementing and research part

- How much data is used when sending sensor data to the cloud server?

- Can data be collected both from the accelerometer from smartphone and medical grade sensors at the same time, and how does this affect the performance of the mobile application?

- How should the wearable sensor and internal sensor be configured to provide data in the same sampling frequency?

1.3 Limitations in this project

The mobile application is developed for android operating system. The reason is that Iphone operating system has stricter restrictions regarding tasks that are running as a background service. This is a required function for the system as it is supposed to collect data over a long time. Also the API that is required for connecting the medical grade sensor is only provided for android.

Limitations for the android application

- Lowest Android version that are supported is Ice Cream Sandwich 4.0.1, some functions that are used doesn't support a lower version

- The phone must be able to sample data from internal sensor at minimum 20Hz as defined by the Gait Algorithm in order to guarantee clinically relevant results.

- The phone must have Internet connection for the gait analysis to work as the analysing part is done in the cloud

- The phone must have built in accelerometer sensor, some phone model doesn't have a built in accelerometer sensor and then the application will not work

1.4 Demarcation between the mobile and cloud projects

This project is divided into two parts. The first part is the mobile application which this thesis is about and the second part is the cloud server. The mobile application handles the collection of data from sensors, it also guarantees that no data points are dropped or lost. After the data has been sent from the mobile application the second part takes over, the cloud server. The main task there is to do gait analysis on the data and visualize the analysed data in a web browser.

2 Background

2.1 Related work

Gait analysis that are using accelerometer sensor has been proposed[10] however a fully working platform containing a mobile application as a gateway that both gathers internal accelerometer data and external sensors data and a cloud server that processes the data has not been discussed. Different approaches has been made and most of the research use wearable sensors as the collector of accelerometer data and not a mobile phone, some of the research that aim for the same result as this project will be discussed in this section.

S. Del Din et al[11]. are doing research on how to make gait analysis more accessible in home environments and for longer periods of time. Their goal is to create a large scale framework for large clinical application. What they suggest is very similar to what this project want to achieve. The big difference is that it is only research on how it could be done and not implemented into an actually framework, though they talk about important aspect of such frameworks like how long time analysis could be done and how free living assessment offers potential better analysis result then compared to clinical testing.

Their conclusion is that using low cost sensors provides rich information about the subjects but there is a lack of integration between different platforms.

As mentioned in [12] a test with ipad to gather accelerometer data for gait analysis was made on 34 subjects and the results showed that the tablet was highly comparable to motion capture systems. The study also shows that body worn sensors was relatively unobtrusive and well accepted by the users. The important conclusion is that low cost and highly available gadgets(tablets, smartphones) can be used to produce valid clinically meaningful data.

[12] also write that "Though continuous assessment in the home may be the “holy grail” understanding measures under controlled settings is needed before the move to uncontrolled settings and remains a major challenge of this eld." which is a major challenge in systems like this because the framework in this project aims for the user to use this in daily life and not in a controlled environment.

Oresti et al[13]. aims to develop a framework for mobile applications that collects sensor data from the internal sensors and from wearable sensors. Their project is similar to what the the aim is for this project, to create a mobile application which can handle different sources of accelerometer data and send it to a cloud server that process the data. The different is that this project goal is to deliver a complete solution from the mobile application to the endpoint(web page that plots the result).

Moving away from the idea of a general purpose research platform, the work of Pan et. al. [14] introduces PD Dr. This work narrows the focus and concentrates on patients suffering from Parkinson's disease. Realizing the value in monitoring subjects or patients in their daily life, the work of Pan presents a platform for gait assessment that allows the patient to conduct tests at home using an app which they call PD Dr. The app presents the patient with a set of tests and instructions on how to go about in performing them. The app uses the embedded accelerometer of the phone to collect data. After a test is completed, the data is sent to a cloud server for processing. The resulting output is sent back to the client and can also be accessed by a doctor via a database. The cloud component of PD Dr is modular in design and is argued to scalable. But according to the authors, the system is not designed to perform long term continuous monitoring of subjects.

The approaches on the literature indicate that to get the better results from gait analysis, subjects need to be monitored in their natural environment and ideally also during longer periods of time. Understanding that studies on large scale becomes resource intensive due to the amount of manual labor needed there is demand for a platform to overcome these problems.

Making use of personal devices such as smart phones helps in further lowering of thresholds for large scale gait analysis, this is demonstrated with the mobile application such as mHealthDroid and cloud service PD Dr. While PD Dr allows for 'at home' gait assessment it lacks the general purpose of the e-Science central and long term monitoring capabilities. The gap this project will fill is to take the idea from a concept to a proof-of-concept platform. This would show that such a platform is possible to develop and that analysing gait is possible with the use of cheap accelerometer sensors instead of expensive lab equipments.

3 Theory

This section explains the different information and purpose of the technology that are used in the project, technologies and terms such as programming language, API, database, Internet of Things, inertial sensors, cloud server, communication protocols and CPU. The mobile application is built in Android studio and written in Java. To be able to connect to the external shimmer an external API is used. If there is no Internet connection on the mobile application there must be a local database that saves accelerometer data locally as it is crucial that no data is dropped or lost on the way. When there is connection again and the smartphone is sampling, the data is sent to a cloud server that process the data. The accelerometer data are being sent with a Internet protocol that is called websockets and allow fast communication over the Internet.

3.1 Programming language

3.1.1 Java

Java is a class-based, object-oriented language that is a good language to start coding in, it was developed by SUN Microsystems in 1995[15]. A big feature java has is WORA(Write once run everywhere) which means that compiled code runs on any platform that supports java without the need for recompilation. The security is based on public key encryption which makes it a secure language to code in, it also provides multi-threads which enables the application to do several operations simultaneously. With multi-threads applications can run more smoothly and be more interactive.

Java has built in garbage collection that makes sure that memory is being released when the object is no longer needed or is in use. In theory this will prevent memory leaks but that is not always the case as objects can still be stored in containers and therefor java compiler don't release the memory it uses. The garbage collection make it easier for the developer as it removes the need for manually freeing memory[16].

3.1.2 Android studio

Android studio is the official IDE for android and was released in 2013[17]. It supports drag and drop for layout creation which is a good UI and structure for projects. It has a built in emulator that enables the developer to run code in a emulated phone directly in the IDE, this makes it easier to debug code as it doesn't need an external phone to run the code.

Android studio has an integrated debug tool that is called DDMS(Dalvik Debug Monitor Server) that are used to debug and benchmark the mobile application[18]. The key features DDMS provides are

- Tracking memory allocation of objects
- Viewing heap usage for a process
- Using the Network Traffic tool

It is important to debug the mobile application to have good performance,reduce the battery usage and make sure there is no memory leaks.

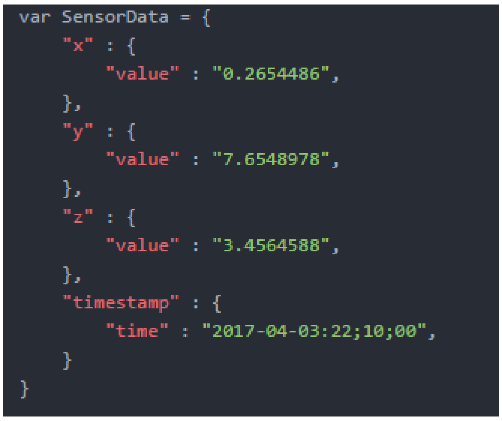
3.2 API

API(Application program interface) is a set of routines, protocols and tools for building applications. It defines methods how to communicate between various software components[19]. A good API makes it easier for programmers to write complex applications as the developer can access other companies data and services like twitter, facebook, amazon and google. APIs provides building blocks for developer. There are several different APIs for operating systems, software or websites.

For example, weather.com has an API that enables app/web developers to access weather information from their database without entering their webpage[20]. If no API would exist the application would have to enter the webpage and read the information directly from the page like humans do. With an API the application sends a message to weather.com that he need information from their database and then get a message back with structured information in JSON format.

3.3 JSON

JSON is a minimal, readable format for structuring data. It makes the data readable for human eyes and easy to access[21]. A example how the JSON structure looks like can be seen in Figure 2.



```
var SensorData = {
    "x" : {
        "value" : "0.2654486",
    },
    "y" : {
        "value" : "7.6548978",
    },
    "z" : {
        "value" : "3.4564588",
    },
    "timestamp" : {
        "time" : "2017-04-03:22;10;00",
    }
}
```

Figure 2: JSON stucture

The JSON object is called SensorData and has four parameters. X, Y, Z and a timestamp. As the data is structured in a organised way it is easy to read and get the data from the object.

3.4 Database

A database is a collection of information organized in a way that makes it easy for computer to access and save certain information in a quick and structured way[22]. Most databases contain multiple tables which includes several different columns(field) and rows(records) where each column contains a specific attribute and each row contain values for the specific attribute. The number of columns in a single database depends on how many different categories of information that needs to be stored, the rows is the quantity of information in each different categories.

Table 1 illustrates a simple database structure. The different attributes like name and surname is the different columns and the attributes to the different columns is the rows. This allows for a computer program to quickly select and handle different pieces of information.

Table 1: Database example

Name	Surname	Day	Month	Year	Tel	Cel
Tim	Svensson	6	November	1989	0431351615	07632504205
Victoria	Svensson	21	October	1991	0431351615	07632504206
Christer	Larsson	5	Januari	1951	0434351516	07632504207
Lena	Karlsson	11	September	1996	0434351555	07632504208

To be able to communicate between databases and computer programs a database management system(DBMS) is needed[22]. A DBMS generally manages the data, data format, field names and file structure. Another important aspect of the DBMS is that it also defines rules to validate and manage data.

3.5 Internet of Things

As Internet gets more widely available and the cost to be connected decreases, more devices are being created with wi-fi capabilities and sensors being built into the devices. The Internet of thing concept is that all devices that has an on and off switch can be connected to the Internet. This includes devices like cellphone, fridge, lamps and wearable devices. By 2020 the analyst firm Gartner says that over 26 billion devices will be connected[23].

The concept Internet of Things was discussed as early as 1982, with a modified coke machine to become the first Internet connected appliance. The machine reported when newly drinks were added and when the drinks were cold[24]. The phrase IoT actually started as Ubiquitous computing, with the meaning of computing to appear anytime and anywhere[25]. In contrast to desktop computing, ubiquitous can occur using any device or any platform, like a fridge, glasses and lamps.

3.6 Inertial sensors

3.6.1 Phone accelerometer

The Android device provides several sensors, such as the accelerometer that is being used in this project.The accelerometer sensor in a phone is a circuit based on Microelectromechanical System(MEMS), it senses the forces of acceleration or movement of the device.It also estimates the angle the mobile is being held at. The sensors are useful for monitoring device movement, tilting, shaking or swing.

Accelerometer measures the acceleration applied to the device (Ad). It does so by measuring forces applied to the sensor itself (Fs) using the relation:

$$\mathbf{Ad = -\sum Fs / mass}$$

In particular, the force of gravity is always influencing the measured acceleration:

$$\mathbf{Ad = -g - \sum F / mass}$$

The sensor gives a multidimensional array with three different values for the coordinate axes x , y and z. The unit of the values are m/s^2[26].

3.6.2 Medical grade sensors

There are a variety of different medical grade sensors that can collect data and provide the data to different devices such as a smart phone. Sensor like Shimmer sensor device can collect sensor data like accelerometer and gyroscope[27]. The shimmer device is a small wearable sensor that sends the collected accelerometer data over bluetooth or saves it to a SD-card. To be able to connect to the device they have several solutions, A mobile application, pc client and API.

3.7 Cloud server

Cloud server work in the same way as a physical server but the functions can be different[28]. When using a cloud server client rent a virtual space on a server rather then renting or running physical servers. There are two forms of cloud server, shared hosting and dedicated hosting. The difference is that shared hosting is a cheaper solution but the server is shared between several clients. The dedicated hosting means that just one client rent the whole server and don't have to share it with other clients.

With cloud hosting resources can be scaled up or down depending on how much traffic there is on the server, this makes it flexible and a cost-effective solution. When there is a high demand on the server the capacity can increase automatically to reach the demands. Cloud servers offers more redundancy then a local server, if one server fails the other will take its place. The key benefits for using cloud server is flexibility, cost-effectiveness, easy to set up and reliability.

3.8 Communication protocol

The communication between the application and server should be as reliable and fast as possible for the user to experience the system as real-time as possible. The Internet protocol that is reliable that is going to be used is websockets which is built on top of the protocol TCP[29]. Websockets enable real time bidirectional communication. It works on many platforms(such as mobile phones and web browsers), and is fast and reliable.

The first time it connects to the receiver it utilize a handshaking. When that is done it can send and receive messages instantly between the two devices that are connected. It works in both directions so the mobile application can both send and receive messages.

3.9 CPU

CPU(Central Processing Unit) is often called the brain of a computer[30]. It is in the CPU most of the calculations take place. In the terms of computing power, it is the most important element of a computer. CPU handles all the instructions it get from hardware and software running on the computer. When measuring the CPU usage of the mobile application there are two different modes, the kernel mode and the user mode.

Kernel Mode
In kernel mode the executing code has complete access to the underlying hardware[31]. It can execute any CPU instructions and reference any memory address. Kernel mode is most often used for the lowest-level, must trusted functions. If there is a crash in kernel mode it can crash the whole operating system.

User Mode
The code that executes in user mode has no way to directly access hardware or reference memory. Code running in user mode must go trough system API's to access hardware and memory. In this mode a crash is recoverable and doesn't have such high impact as when a crash occurs in kernel mode. Most of the code in the mobile application will run in user mode.
As seen in Figure 3 the kernel mode is in the center and is most privileged. In the outer

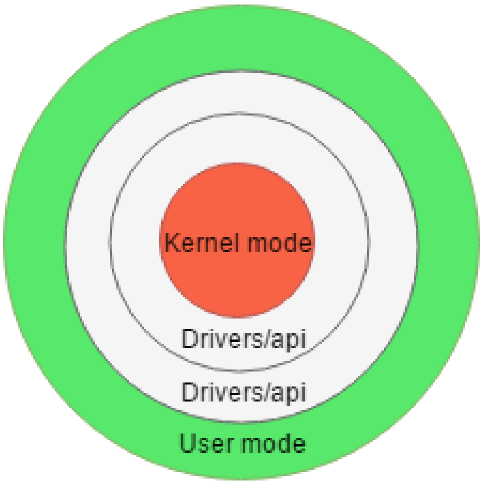


Figure 3: Kernel and user mode

end the user mode is and is least privileged. If the user need to send data or get data from kernel it must go through driver/API layers.

When measuring the mobile application with DDMS it shows both how much CPU the kernel and user mode uses. The phone has two settings regarding which sensor to use, the internal and the external sensor. The CPU usage during collecting from one sensor or both sensors will be analysed.

4 Proposed Methodology

The focus for this part of the project is to develop a mobile application that collects data from two sources of data, the internal accelerometer in the mobile phone and a medical grade sensor. The data that is collected is sent to a cloud server which does gait analysis on the collected data. The user interface of the mobile application is going to be easy to use. A goal is also to structure the code in a way that enables other sources of sensors to be added in the future.

4.1 Project structure

The project is divided into three parts, implementation, research and education. The research part is about reading related work, writing the thesis and do research on how to solve complex problems. The implementation part is about coding the mobile application and testing the reliability of it. The education part involves studying how to code and how to structure a mobile application and learn to use tools like Github[32] and Android Studios[17].

The project is done using agile scrum method where blocks of coding are done in sprints. A sprint is typically 1-2 weeks. After a sprint is finished a evaluation of the code and testing is done. The code that has been implemented is tested in a way to show that the code is working as intended. Trello[33] is used to do the scrum model. Also if there are questions between the groups Slack[34] is used as a communication forum.

4.2 Version control

Github is used for version control, it is being used so that the user is being able to have control over the project and be able to revert back to an older state of the project if something goes wrong. When using Github the data is stored in their database and is easily accessed from different computers.

Github is especially good to use if there are multiple person coding on the same project. There is a master branch which is the official release, and there are branches where new coding is done. When the code in branches is stable and working as intended a merge is done back to the master branch and it get a new official version.

4.3 Choice of platform and programming language

In an early stage of the project the decision was made to only develop the mobile application for smart phones using android as operating system. The data collection part needs to run in the background and even if the screen is off. Other operating system have stricter rules of what can be done if the application is inactive or the screen is off.

With the platform choice of android the programming language is Java. Java has a good library and framework support, it also has a big community. The API to be able to connect a shimmer to the application is only developed for android and that was a major reason when choosing platform as it is a requirement for the mobile application.

4.4 Structure of the mobile application

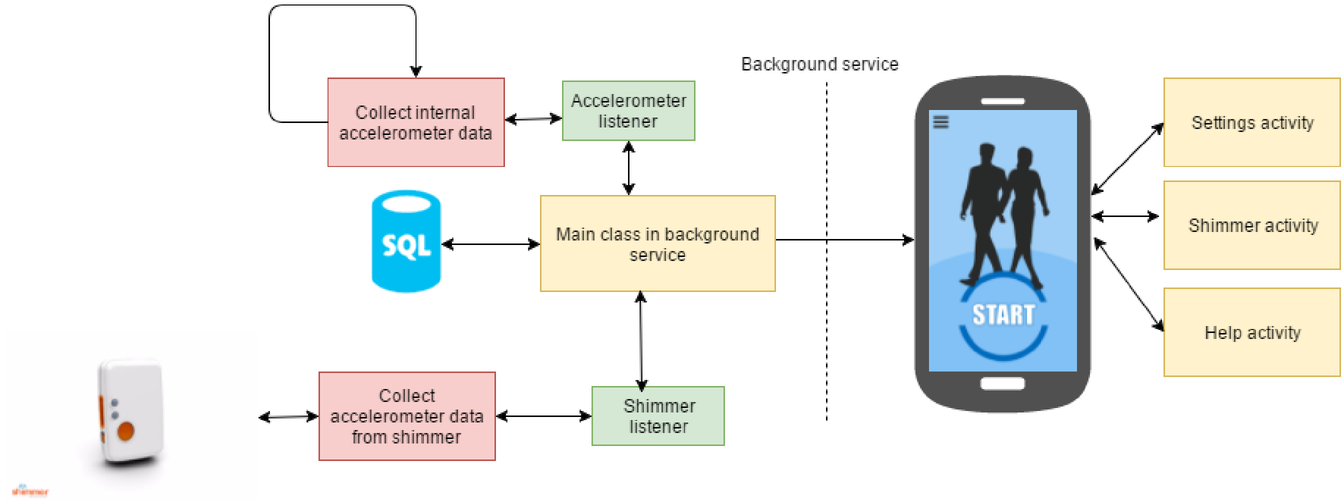


Figure 4: Structure of the mobile application

The structure of the mobile application can be seen in Figure 4. It is built around a pattern called observer pattern. The observer pattern is a software design that maintains a list of its dependents and notifies them automatically if any state changes. The mobile application was built with this pattern so it would handle and process accelerometer data effectively when new data was registered from the sensor. When the user starts the application there is a button to start the background service where the collection of accelerometer data occurs. The background service handles the functions that requires to run until the application is stopped. These function are:

- Collecting accelerometer data from the inertial sensor and the medical grade sensor
- Saving the data to the database
- Sending data to the cloud

The goal is to build the whole process that runs as a background service as a framework. It can be exported as a library and used in other application that requires collection of accelerometer data from inertial sensor and medical grade sensor. To be able to do this the code must be structured in an easy and good way so that other people who use the framework understands it.

There are two listeners that listen if the sensors register new values. The listener class is a interface with one callback method attached to it. When a new value is registered from a sensor the callback method is called in the listener and the main class that listen for the callback get the new values.

If there is no Internet connection the data cannot be sent to the server for the analysing. The data is then stored in a SQLite[35] database directly on the mobile phone. When there is Internet connection again data is taken from the database and starts to being sent to the server.

Apart from the background services there are some activities that happens on the user interface thread.

- Settings Activity
- Shimmer Activity
- Help Activity

These activities requires user interaction and therefore they cannot be in the background service. In the setting activity the user defines how many sensors that collects data, at which sampling rate and where the sensor is placed on the body. In the shimmer activity the user can see which external shimmers that are connected to the mobile phone. If there are multiple shimmers connected the user choose which one to use. The help activity give the user a brief explanation how to use the application and information about the sensors.

4.5 Collect and process accelerometer data

As mentioned in section 4.4, there are two listeners that listen for registered data from the sensors. When a new sensor value is registered the class informs the listener and sends the value to it. Then the listener tells the main class that there is a new value that is available and sends the value. When the main class receives the value it packages it in a JSON-object. JSON-object is used as it is simple to send them with the use of websockets. The values that are in the object are a timestamp when the value was registered, x angle, y angle and z angle. A index value is also stored in the JSON-object to keep track of how many samples that have been collected. The structure of the JSON-object can be seen in Table 2

Table 2: JSON structure

JSON structure					
Index	Channel	Timestamp	X	Y	Z
Index value	1 or 2	The time when the data was collected	X value	Y value	Z value

4.6 Send data to server

There are several protocols to send data over the internet. The protocol that this project use is websockets. Websockets allows data to be sent fast and without acknowledgement which suits the need to send data fast bidirectional. When deciding which protocol to use it was between Websockets and REST. REST is based on HTTP which has operations like get,post,update and delete[36]. For each package that are sent with REST a port is opened and when the message has been delivered the port is closed again, that makes the overhead bigger and more data is consumed. For real time transfer or streaming of data REST is not the best suited protocol.

When a connection is made to the server a handshake is needed. To successfully do this handshake an JSON object is created that is called register, there are 5 properties, the name of the user, the type of the user(publisher), the sampling frequency of the sensors, how many channels(one per sensor) and where the sensors is on the body. The user defines the username, sampling frequency and where the sensors are in the settings activity. The JSON object together with an event that is called register is then sent to the server and the handshake is established. When the handshake is established data can be sent bidirectional. Below is a snippet of code how the registration is made to the server.

```
public void register(String _type,int _channel,
   double _samplingFreq,
String _shimmerPlacement, String name)
```

First the method register is called. The in parameters are what type of user that requires registration, in this case a publisher, the channel which tells the server how many sensors that are connected, the sampling frequency of the sensors, where the sensors are attached on the body and the name of the user.

```
JSONObject connect = new JSONObject();
```

A JSON object called connect is created

```
connect.put("type", _type);
connect.put("user",name);
connect.put("freq", _samplingFreq);
connect.put("channel", _channel);
connect.put("place", _shimmerPlacement);
```

The different properties that was required for the registration are packed into the object

```
mSocket.emit("register",connect);
```

And the object is sent to the server for registration. When registration is done data can be sent instantly when a new sensor value is collected.

4.7 Storing data in a database

There is several ways to store data on the mobile phone. The most common way to save data is to save it to a file on the phone, such as a csv file or save the data into a database[37]. The choice for this project was to use a database locally on the phone. That makes it easy to get specific data from the database and also easy to save the data, although it is quicker to save it to a text file the database is more structured and easy to handle.

When there is no Internet connection the data that are collected from the sensors needs to be stored in the local database on the mobile phone. The database that is being used are SQLlite. The data is stored in the same structure as it is being sent to server to make it easy to send the data when there is Internet connection. How the data is stored can be seen in the code below.

```
private static final String CREATE_MOTION_TABLE =
   "create table "+MOTION_TABLE+" ("
            +KEY_ID+" integer primary key autoincrement,
               "
            +TIMESTAMP+" real, "
            +INDEX+" real, "
            +ACC_X+" real, "
            +ACC_Y+" real, "
            +ACC_Z+" real); ";
```

5 Results

The result of the project is a mobile application that can use the internal accelerometer and a external shimmer to collect accelerometer data. As long as the smartphone has Internet connection the system works anywhere in the world. It can collect from one of the sensor or both at the same time, when using one sensor symmetry between the legs can not be measured. To get the best result from the analysis two sensors are recommended.

In the result section four major topics will be discussed and analysed.

- **Bandwidth** - The amount of data that is being sent and how that can be optimized

- **Performance** - How much power the application consumes

- **Synchronization** - How the synchronization between the sensors are working

- **Design** - The design choice for the application

5.1 Bandwidth

For the real time aspect of the application data is sent to the server immediately when new values are registered from the sensors. This enables data to get to the server fast but has a drawback in consuming more mobile data. Another approach would be to collect data into chunks and send them with a given time interval, both options will be analysed in this section.

5.1.1 Sending each registered values

When sending a single message using websockets protocol the overhead is approximately 74 bytes[38]. For each message that is being sent the overhead is being added to the message. If the sampling rate of the sensor is 120Hz the overhead is 74 B (Byte) * 120 values/s. The total amount of overhead is then 8.8 Kb (Kilobyte) after sending each new value to the server during one second. The maximum data that fits in one message is 1426 B (1500 B - header), this is called the maximal transfer unit(MTU). If the message is bigger it is being split and sent as several messages.

Example:

A message is 1 Mb (Megabyte) = 1,000,000 B . Each message can be of maximal 1460 B.
1,000,000/1486 = 685 messages.
For each message there is a overhead of 40 B.
40 B * 685 messages = 27,400 B = 27,4 Kb. The total amount of data that actually is transmitted are 1,000,000 B + overhead 27,400 B = 1,027,400 B.

A JSON object that contains index value, timestamp, x, y and z values are sent to server using websockets. The amount of messages that are being sent is dependent on the sampling rate of the sensor. If the sampling rate is 120Hz then 120 messages are sent each second. To be able to calculate the size of the JSON-object a simple server was written in node.js. The server receives the object and calculate the size of it.

When using this method it shows that the size of the JSON-object when using one of the sensor is 48 B, the size are the same independently which sensor that are being used because the same parameters are used in both. When using both sensors the size of the object is doubled as can be seen in Table 3. The total amount of data the mobile application sending for each sample is 48 B + 74 B (overhead) = 122 B.

Table 3: Size of JSON-object

Sensor	Size of JSON-object
Internal sensor	48 B
Medical grade sensor	48 B
Both sensors	96 B

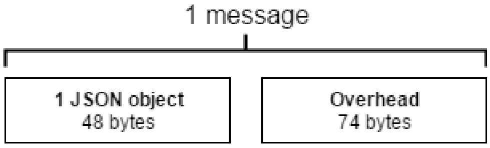


Figure 5: Sending each registered value

Example of 1 hour usage with 1 sensor and 120Hz sampling rate:

120Hz sampling rates means 120 JSON objects are created and sent each second.
120 JSON objects * 122 (size of each JSON object) = 14640 B/s (Bytes per second). In one minute its 14640 B * 60 s = 878400 B/m = 0,8784 Mb/m. After 1 hour collection of data the amount of data that has been sent is 0,8784 Mb * 60 m = **52,7 Mb/h**.

If the sampling rate is lowered to 50HZ the amount of data that are being sent decreases.

Example of 1 hour usage with 1 sensor and 50Hz sampling rate:

50 JSON objects * 122 (size of each JSON object) = 6100 B/s. In one minute its 6100 B * 60 s = 366,000 B/m = 0,336 Mb/m. After 1 hour collection of data the amount that has been sent is 0,48 Mb * 60 m = **21,96 Mb/h**.

As seen in the examples the amount of data that are being sent to the server decreases significantly when lowering the sampling rate of the sensors, but the quality of the analysed data gets lower. To get the best result its recommended to use wi-fi and sampling rates above 120Hz. When using two sensors the data doubles.

5.1.2 Sending chunks of data

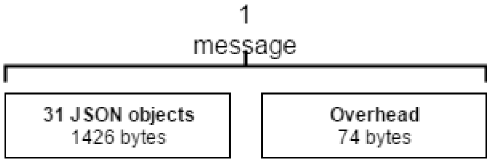


Figure 6: Sending chunks of data

Instead of sending the data to the server each time a new value get registered the data can be stored in chunks and sent after a given time. When sending in chunks the total amount of mobile data usage is lower as more values can be stored in one message and the amount of overhead is fewer. One message can hold 1426 B. With each JSON object being 46 B a total of 31 JSON objects can be stored in one message.

When sampling for one minute at 120Hz the total amount of messages that would be sent are 31 instead of 120 when sending each new value. When sending in chunks the total amount of data being sent are 6000 B/s as 4 messages are sent during 1 minute. When sending each new value the total amount where 19200 B/s. The amount of mobile data that are saved when using chunks are 19200 B/s - 6000 B/s = 13200 B/s. When sampling for one hour the total save of sending chunks instead of single values are **47,52 Mb**.

5.1.3 Testing the bandwidth with Data Monitor Usage

The network usage can be measured with a tool called Data Usage Monitor[39]. It allows the user to measure data that are being sent during real-time. According to the calculations when sending each new value to the server with the sampling rate 120Hz the total amount of data that are being sent each second is 14640 B = 14,64 Kb. To test this numbers Data Monitor Usage is being used. The result can be seen in figure 7. The two arrows in the top left corner shows how much data that are being sent and received each second.

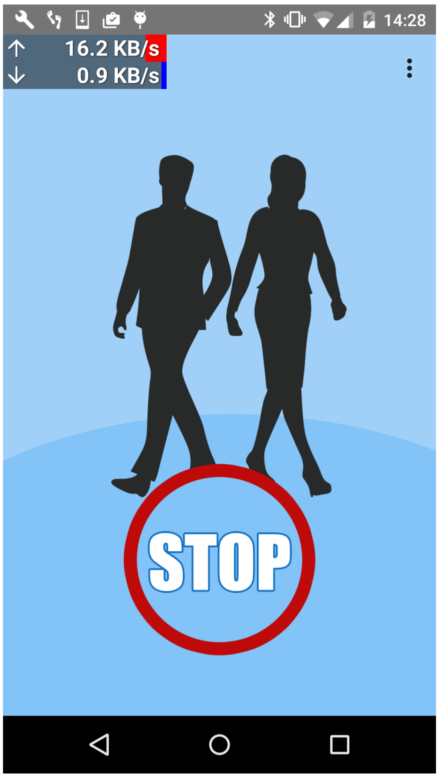


Figure 7: Mobile data usage

When measuring with Data Monitor Usage the application sends approximately 16 Kb/s to the server when sampling at 120Hz. According to the calculations in 5.1.1 the real data usage is very close to the theoretically data usage. The error is (16 Kb/s - 14,64 Kb/s)/16 Kb/s * 100 = 8,5 %.

5.2 Performance

The application is supposed to run under long periods of time so the performance aspect is important. For the mobile quality assurance one of the most critical concern is the user and how the user experience the mobile application. Three components that will be analysed are CPU, memory usage and battery consumption.

5.2.1 CPU

Using one sensor - Internal sensor
The CPU usage when only using the internal sensor with 120Hz sampling rate can be seen in figure 8

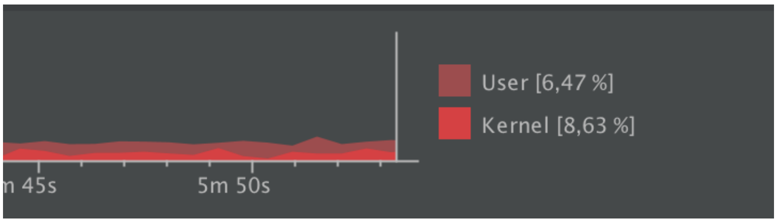


Figure 8: Internal sensor CPU usage

The usage is approximately between 5 - 10 % in the user mode. Sampling data from the internal sensor is not computationally heavy and doesn't affect the performance on the mobile application significantly. When lowering the sampling rate the CPU usage decreases. For example, when lowering the sampling rate to 80 Hz the CPU usage is approximately between 2 - 8 %. The kernel mode is between 3 - 9 % as it uses android built in trusted functions to get the values from the sensor.

Using one sensor - Shimmer sensor
The shimmer sensor uses more CPU then the internal sensor when sampling at 120Hz, the usage can be seen in figure 15

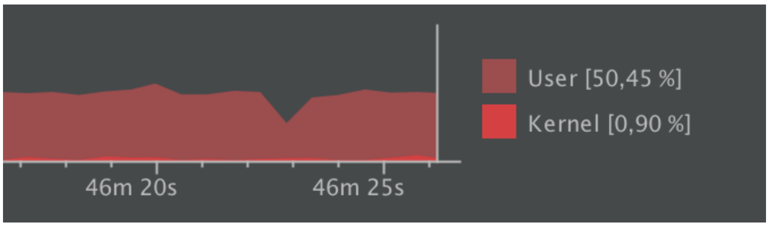


Figure 9: Shimmer sensor CPU usage

The CPU usage is much higher when using a shimmer sensor instead of the internal sensor. It uses more CPU because it is a API that handles the bluetooth connection to the shimmer and the collecting of accelerometer data from the sensor. The kernel mode doesn't use consume much CPU because it's not a android API and therefore not a trusted one.

Using two sensors - Shimmer sensor and Internal sensor
When using both the sensor both the user mode and kernel mode consumes more CPU. The collecting from the internal and from the shimmer is occurring simultaneously. The total CPU usage can be seen in figure 10

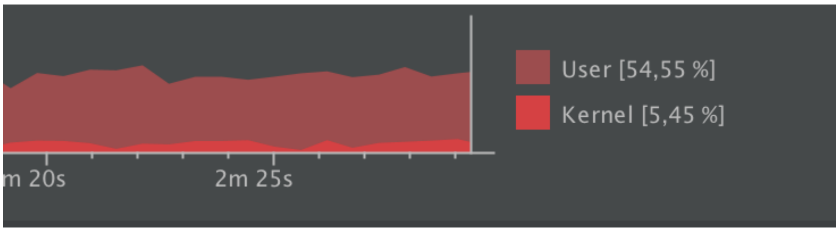


Figure 10: Both sensors CPU usage

5.2.2 Memory usage

To make sure that the application doesn't crash during collection of data memory needs to be tested. If there are memory leaks this could lead to application crash when the memory is full. To test the memory DDMS is used in android studio. There is a real-time memory analysing tool that present to the user how much memory that are being used and how often the garbage collector frees memory. A example how it looks like when analysing the memory can be seen in figure 11.

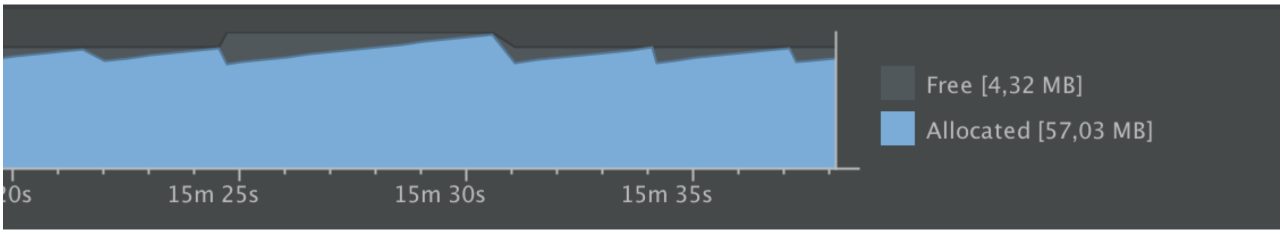


Figure 11: Analysing memory after 15 minutes using both sensors

To make sure that there are no memory leaks the allocated memory should not increase over time. In figure 11 the application has been collecting data for 15 minutes from the shimmer and the internal accelerometer at a sampling rate of 120Hz. The allocated memory has not been increasing over time and that is the result of not having any memory leaks. The garbage collector finds all the objects that doesn't have any references and frees the memory. The memory was also analysed after 30 minutes to and the result can be seen in figure 12.

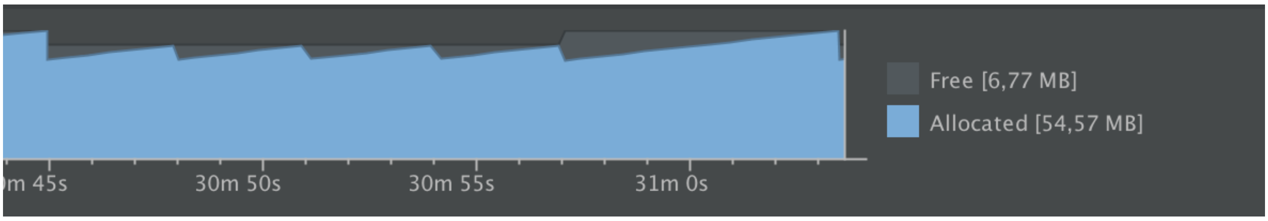


Figure 12: Analysing memory after 30 minutes using both sensors

After 30 minutes collection of data from both sensors at 120Hz it still has the same allocated memory. To make sure there is no memory leak when using one sensor the test was also made when using only the internal sensor at a sampling rate 120Hz. When the application had been collecting for 15 minutes the allocated memory was the same as when the application started. The result can be seen in figure 13.

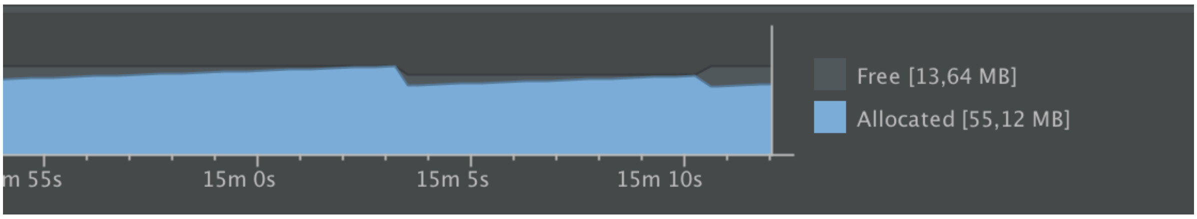


Figure 13: Analysing memory after 15 minutes using internal sensor

The allocated memory doesn't increase and the garbage collector finds all the objects that there are no references to. To be sure of the result the memory was analysed after 30 minutes collecting of data. The result can be seen in figure 14

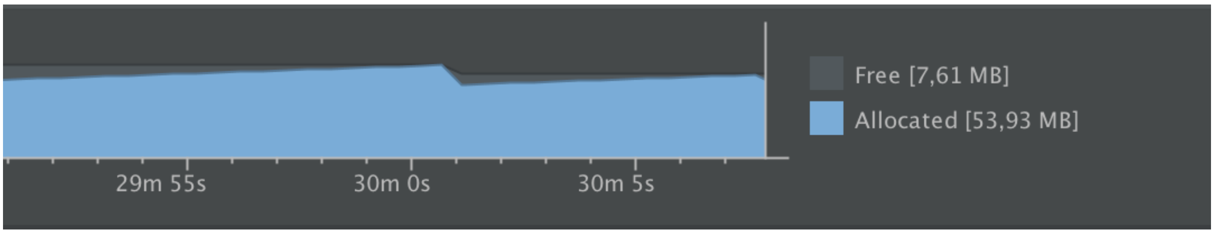


Figure 14: Analysing memory after 30 minutes using internal sensor

The last memory test that was done was when using only the shimmer sensor. The test is after 15 minutes collecting of data at 120Hz as seen in figure 15.

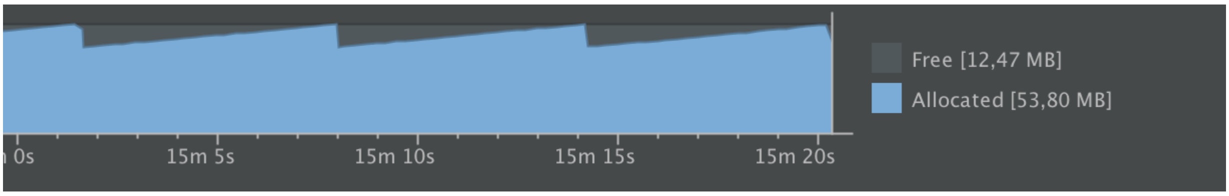


Figure 15: Analysing memory after 30 minutes using internal sensor

The tests indicates that there are no memory leaks. Independently of which sensor that is being used the garbage collector finds the objects that has no references and releases the memory. If the allocated memory would increase over time it would lead to failure of the mobile application when the memory is full and eventually crash it. The application is supposed to run during long periods of time so it is important that there are no memory leaks.

5.3 Battery consumption

The battery consumption of the mobile application are tested with a test. The test is to fully charge the phone and then send data from both sensors at a sampling rate of 120Hz and see how long the battery last. The phone that are being used in this test is LG Nexus 5. The phone's battery size is 2300 mAh.

5.3.1 Battery duration during collection of data

Data was collected from both sensors and the screen was turned off on the mobile phone. No other programs were running in the background. Before the test the phone was fully charged to 100% and the test was ongoing until the battery was empty. The result can be seen in Table 4.

Table 4: Battery consumption

Time (minutes)	Battery %
0	100
19	90
40	80
62	70
85	60
111	50
141	40
169	30
197	20
215	10
245	0

The total time the mobile was able to collect data and send it to the server is 245 minutes (4 hours and 5 minutes). When plotting the numbers in a graph it becomes clear that the mobile consumes equal amount of battery over time as the line is linear.

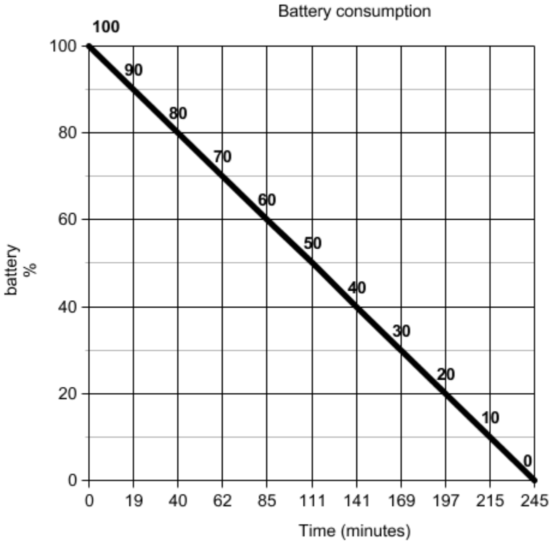


Figure 16: Plotting the result from battery consumption test in a graph

5.4 Synchronization

A big aspect when collecting data from two different sensors is to synchronize the data so that the server get the data from both sensors at the same time. The internal sensor is not as accurate as the shimmer sensor when setting the sampling rate. Even if the sampling rate is set to 120 Hz the real sampling rate is not 120Hz on both sensors yet it is close. If there would be no synchronization the two sensors would get a increasing offset over time as the server uses index from sensors to determine how many data points it has received. If one sensor register more data than the other the index will increase faster and create the offset.

The solution for this problem is to let the sensor with the lowest sampling rate decide which data points that are sent to the server. In this case it is always the mobile phone that has the lowest sampling rate. Each time a new data point is registered from the shimmer it is saved in a shimmer object. When the sensor in the mobile phone register a new value it takes the latest value in the shimmer object and send values from internal and the shimmer sensor simultaneously.

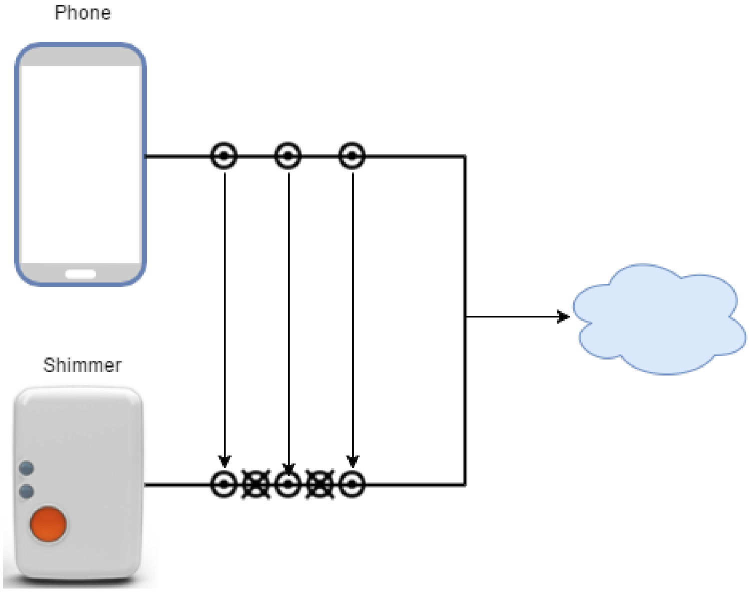


Figure 17: Synchronization between the phones sensor and the shimmer sensor

As can be seen in Figure 17 the phone has the lowest sampling rate and decides at what rate the data is sent to the server. The disadvantages with this approach is that some data points are lost from the shimmer but the method guarantees that there will be no offset. The optimal approach would be to find another solution like re sampling the phone so that both the internal sensor and the shimmer sensor have the exact same sampling rate.

The drawback for this method is that some values from the shimmer sensor are not used. To understand how this affects the signal two graphs will be presented. One is the signal from the phone and the other is from the signal from the shimmer. Both are sampling at 120Hz and the synchronization approach where the sensor with the lowest sampling rate decides which data points that are sent.

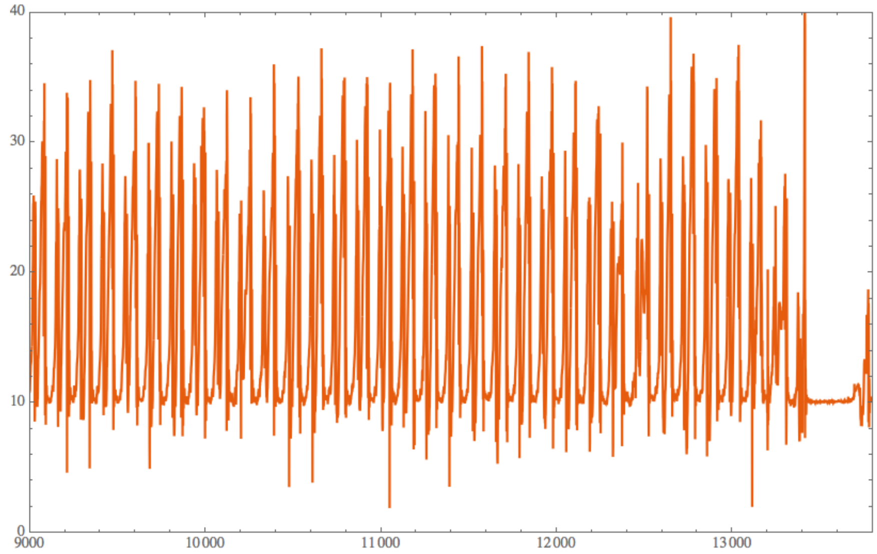


Figure 18: The mean value of the signal from the phone

In Figure 18 is the signal when sampling at 120Hz from the phone. The formula to get composite signal from the accelerometer data are: $\sqrt{x^2+y^2+z^2} = Signal$.

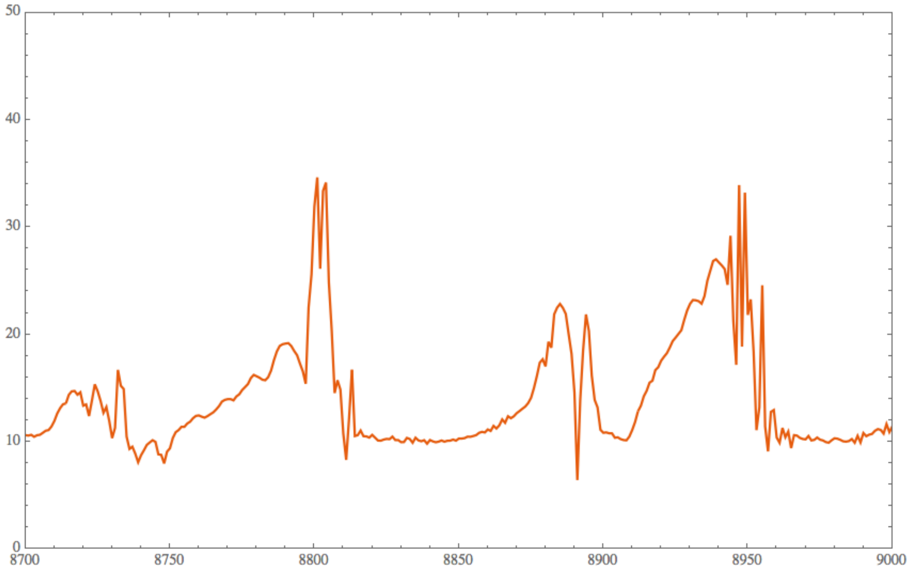


Figure 19: Zoomed in on the phones signal

Figure 19 is the same signal as Figure 18 but zoomed in on the signal. The interesting part is when looking at the shimmer signal.

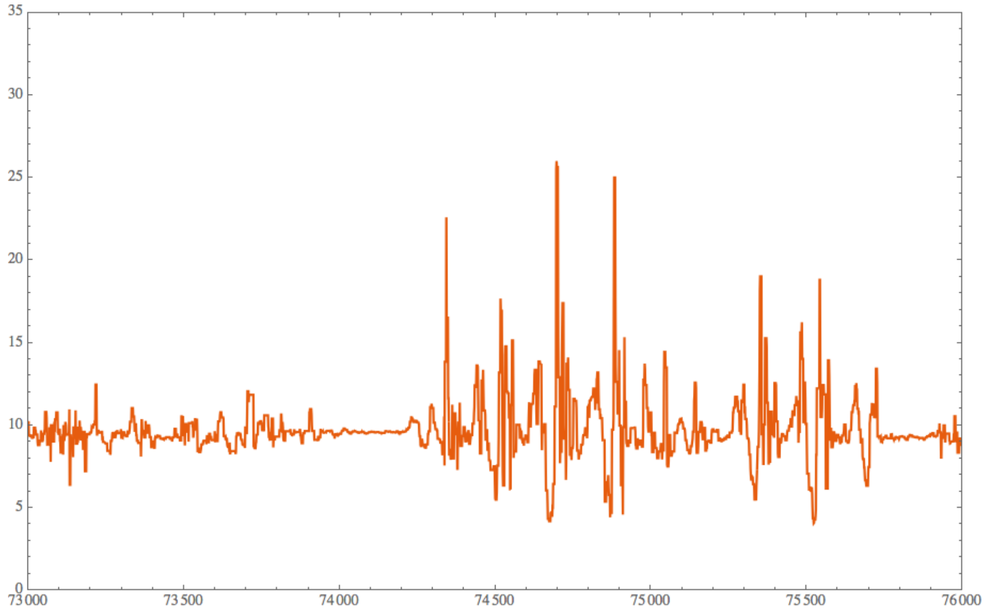


Figure 20: The mean value of the signal from the shimmer

In Figure 20 is the signal when sampling at 120Hz from the shimmer. At first the signal looks good but when zooming in on the signal the difference can be seen.

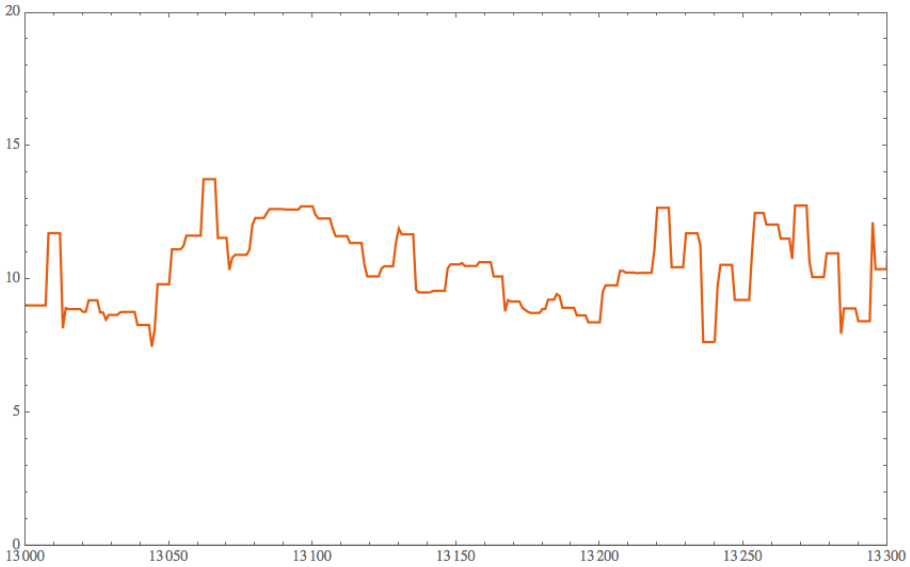


Figure 21: Zoomed in on the shimmer signal

Figure 21 shows the signal from the shimmer zoomed. Here the drawbacks of the synchronization approach can be seen. As some of the data points are dropped the signal gets cut off at some points. The high peeks in the signal occurs fast and therefore it's easy to miss them when dropping data points.u

A solution for the synchronization problem would be to do signal processing, decimation is the process to reduce the sampling rate of a signal so that no aliasing occurs when dropping samples. it is done by first do low-pass filtering and downsampling [40]

5.5 Design

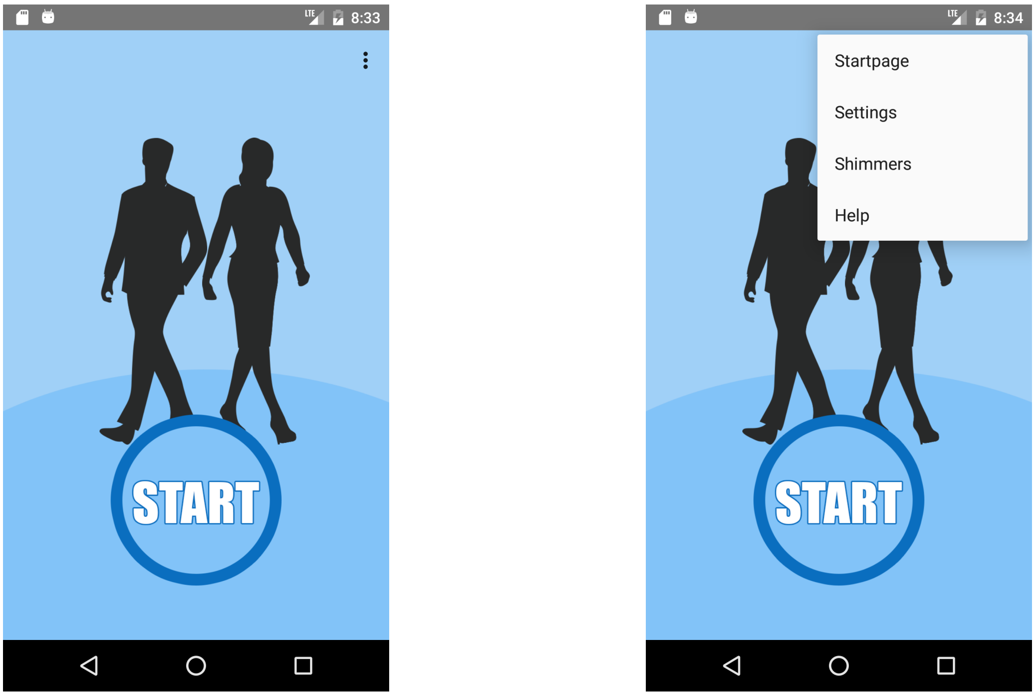
At the planning stage of the design for the mobile application some keywords was decided to build the design around. The keywords where, minimal, easy to use and scalability.

- Minimal - No unnecessary things in the design

- Easy to use - The application should be easy to start and use. The main goal was to be able to start the collection of data with just one press

- Scalability - It should be easy to add new features and sensors

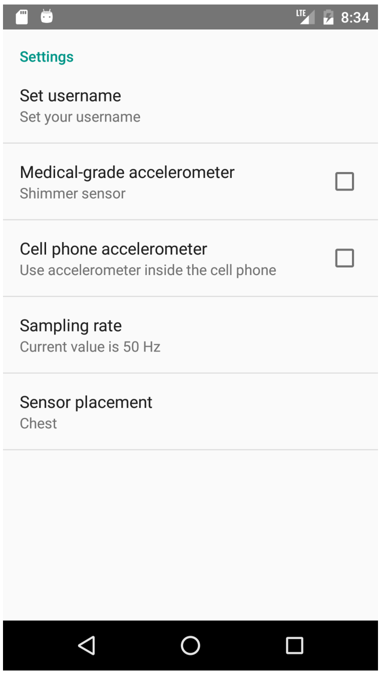
With these keyword a design was developed. The result of the main page can be seen in figure 22



(a) Main page without menu (b) Main page with menu

Figure 22: Main page

The picture of 22a is the main activity that starts when the mobile application is started. Here is a big explanatory button to start the collection of data. In 22b is the same activity but with the menu pressed. The options the user have is to go to settings, shimmers or help. In the settings tab the user can enter a username, which sensor to use, the sampling rate and where the sensors are placed. See figure 23.



Settings

Set username
Set your username

Medical-grade accelerometer ☐
Shimmer sensor

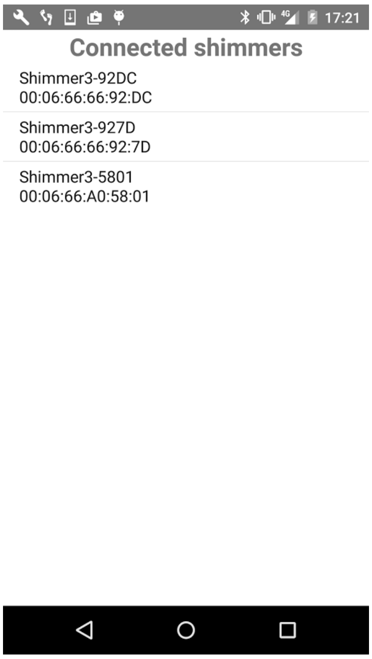
Cell phone accelerometer ☐
Use accelerometer inside the cell phone

Sampling rate
Current value is 50 Hz

Sensor placement
Chest

Figure 23: Settings activity

When starting the application the settings that were active last time it was used are activated and if the user don't need to change any settings they can press the start button instantly when the application is started. The last activity is the shimmer activity, in the shimmer activity the user choose which shimmer to use. All paired shimmers is available here. See figure 24



Connected shimmers

Shimmer3-92DC
00:06:66:66:92:DC

Shimmer3-927D
00:06:66:66:92:7D

Shimmer3-5801
00:06:66:A0:58:01

Figure 24: Shimmer activity

6 Discussion

When the proof-of-concept was developed another architecture of the mobile application was implemented. The new architecture uses a message queue instead of sending every new data point as it is registered from the sensors. The reason why a new architecture was developed was to be able to make sure that no data point was dropped and that they were delivered to the server. Due lack of time the new version could not be properly tested and therefor the first version is the main one and considered as stable. If this project is to be continued it is highly recommended to use the second version as it makes it even easier to add more sensors and to save all data points to a csv file on the mobile phone. The second version don't use a database, instead all data points are saved to a csv text file, the amount of CPU power that are being used are reduced compared to database storage. Also another great feature is that it's even easier to transfer the file to a computer and to import it to different programs, such as Excel.

For future development tests would be required for the second version of the mobile application and also to ability to add different types of sensors. The shimmer API consumes considerable amount of CPU power and that drains battery. To be able to collect data during long period of time it would be optimal to try to write a self made API to get data from the shimmer instead of using the one who already exists.

This project was developed to enable gait analysis for a large number of people and it was crucial to use gadgets that is widely available, such as smartphones. By using gadgets that most people already have in possession the cost of using the platform is very low. Today people have to visit a gait lab to do the gait analysis but that is very expensive and with this platform more people would be enabled to do the analysis with gear they already own or can buy at a relatively cheap price.

When the server is hosted in the cloud several users share the same computers that hosts the servers and therefor it is more effective regarding the environmental aspects. When hosting own servers the whole capacity of the servers are often not utilized and own hosted servers uses unnecessary energy[41]. Also as the sensor data is collected through smartphones no extra hardware is needed for the platform to work.

As this is a proof-of-concept platform no measures was made on the security and integrity part of the mobile application. If the mobile application is to advance the connection and the data that are sent to the server should be encrypted, also the data that is saved in the database on the phone is in plain text and security wise that is not the optimal solution.

7 Conclusion

During this project a mobile application was developed in order to be able to collect accelerometer data from the types of sensors, the internal accelerometer sensor in the mobile phone and from a shimmer sensor. The collection from the sensors occurs simultaneously. The main goal was to develop a proof-of-concept application that shows that it is possible to use low cost sensors such as the internal sensor in a mobile phone to be able to do gait analysis without the use of expensive indoor lab equipments. The gait algorithm that this platform is built around is discussed in a paper[4].

Several tests has been made to make sure the functions and the robustness of the platform is at a satisfied level. These tests shows that the proof-of-concept works and considering the question in section 1.1 the goal has been fulfilled. The goal was to collect data from the inertial sensor and from medical grade sensor to a smartphone over a long period of time and process the data to a cloud server.

In section 1.2 three different research questions was proposed.

How much data is used when sending sensor data to the cloud server?

The bandwidth when sending data are:

Sensor/s	Hz	Bandwidth/Minute (Mb)	Bandwidth/Hour (Mb)
1	50	0,366	21,96
1	120	0,8784	52,7
2	50	0,732	43,92
2	120	1,7568	105,4

Can data be collected both from the accelerometer from smartphone and medical grade sensors at the same time, and how does this affect the performance of the mobile application?

Data can be collected at the same time. When using both sensors the mobile application uses more CPU power, consumes more battery and sends more data to the server. If the user need to analyse symmetry between both legs the precondition is to use two sensors, and if gait analysis is done on one leg only one sensor is required.

How should the wearable sensor and internal sensor be configured to provide data in the same sampling frequency?

The sensor with the lowest sampling rate decide which data points that are sent to the server, in this case the mobile phone. When the sensor in the mobile phone register a new value it takes the latest value from the shimmer and send values from internal and the shimmer sensor simultaneously to the server.

As this can enable gait analysis without the use of expensive lab equipments there are much work left on the mobile application. As this is the first version of the mobile application more optimisations can be done to improve different part of the application such as battery life, CPU usage and bandwidth. Overall this project was a success and a wider experience was gained in how to develop mobile applications in Android.

A weakness in the mobile application is the synchronization when two sensors are sampling simultaneously. The proposed solution in this thesis works but it's not optimal as some of the data points are being dropped. Although a new version of the application was developed there was no time to test it and use that version that has a fix for the synchronization.

8 Appendixes

8.1 Project plan

Project plan

Gait platform

Mobile application
Tim Svensson

Members of the project(platform)

Name	Responsibility	Email
Hampus Carlsson	Developer/Server	hamcar14@student.hh.se
Marcus Kärrman	Developer/Server	marcka14@student.hh.se
Tim Svensson	Developer/Mobile application	timsve14@student.hh.se

School of Information Technology – Halmstad University
Course coordinator: Björn Åstrand, bjorn.astrand@hh.se
Supervisor: Nicholas Wickström,Siddhartha Khandelwal

1. Introduction

Gait is a person's manner of walking, stepping or running.

Gait is normally centered around three main components: locomotion, balance and ability to adapt to the environment. In the human body this is achieved through a balance between various interacting neuronal and musculoskeletal systems[1]. If the function of these systems were to be compromised, effects of this will be apparent in gait analysis. For instance, a disease like Parkinson's disease (PD) will have measureable effects on gait even at an early stage in the disease's progression[2].

The gait cycle is described as the time between successive foot contact of the same limbs. Throughout a gait cycle there are 2 steps and 1 stride. A step is contained within the time of a heel strike of one foot and the heel strike of the other. A stride is completed after two successive heel strikes on the same foot[3]. The most important events in a normal gait cycle are heel strike and toe off, and that is also why being able to detect these events is of great importance in any gait analysis application.

There is some variation in the approach to gait analysis. Labs equipped with motion capture and force plate equipment offers very rich data but suffers from not being able to monitor subjects during longer periods of time[4]. Recent years has seen progress in the field of small and lightweight micro-machined electromechanical systems(MEMS)[9] allowing for approaches to gait analysis which involves wearable sensor technologies[5].
These sensors are typically equipped with accelerometers and or gyroscope. Subjects may be instructed to walk at a given pace in some direction and the analysis is completed by running the collected data through some specialized gait analysis algorithm[6].

Tests confined to labs and dependent on supervision are, besides the problems related to cost, suffering from not being able to observe patients during longer periods of time in daily life. The efforts made into tackling both the problems of cost and monitor time has yielded: (i) better algorithms for analyzing gait in varied environments[7]; (ii) also mobile applications which uses the embedded sensors of smartphones[6]; (iii) remote processing which makes gait analysis available for more people.

There is demand for a framework for performing gait analysis in real-world environments. . Some general purpose platforms for performing scientific experiments have been proposed along the years[13]. However, there remains a challenge regarding integrating applications reliant on a more rich runtime environment, which will typically be the case for gait analysis algorithms. Thus a project which aspires to the develop this kind of platform would have to tackle problems as fundamental as getting these algorithms to run in a cloud environment. Making a platform more narrow and specialized for gait could also allow for a more tailored architecture and targeted optimization.

2. Purpose

Develop mobile application with the goal of supplying the server with accelerometer data for gait analysis like the one discussed in *introduction* to identify and provide solutions for the problems such an endeavour would entail. The work will be centered around the following non-functional requirements reliability, performance and accessibility.

2.1 Limitations

The limitations for the mobile application are

- There will only be a Android version
- It requires the phone to handle at least 40 Hz sampling rate
- The minimum version of android is 4.0.3 Ice Cream Sandwich

3. Method

3.1 Related work

As the use of wearable sensors and analysis of the data they produce is showing to be a valid method[10] of assessing gait, the cost of performing gait analysis can be lowered. With a typical sensor being wearable and not by itself confined to use in a particular space or setting, it is relevant to look into the effects the setting is having on the quality of gait.

According to Weiss[12] and Din[11] performing gait analysis in highly controlled settings is suboptimal. This is said to be because of two main reasons. The first being that the course, track or path along which a subject must navigate will not be as varied as a it would be in a real life scenario. The subject may also receive instructions on *how* to navigate the path, for instance: stop and turn. The second is that the subject is not observed during longer periods of time.

A controlled setting will inherently be less busy in terms of the amount of information the subject has to deal with at a given point in time. Combining this with the increased focus brought not only by the situation itself but also the instructed fashion in which the test is conducted is believed to have the effect of subjects performing better on the tests than they would in a real life setting[11].

For anyone wanting to conduct gait analysis of subjects in daily life there are three main components that must be in place. First, the subject(s) must have access to a sensor, for instance a wearable accelerometer. Second, there needs to be a way for the subject to send the data back to the concerned party, the researcher. Third, the data must be analyzed. While the first part, the distributing of sensors to subjects for use at home is a manageable task, the second and third is considered more problematic[11]. After the sensor has been worn at home by a subject, the data must be sent back to the researcher via some file sharing service like Dropbox™. The researcher then has to run the processing algorithm on each and every dataset received. In their example, a monitor period of a single patient for 7 days will result in 250 mb of data. This would then have to be processed, and would take around 20 minutes for *each* data set to finish. In this instance processing is done in MatLab™

The e-Science central[13] aims at making it easier for researchers to “store, share and analyse their data, and for developers to create new scientific services and applications”. The e-Science central is cloud based and allows for researchers create workflows by combining applications that either already exists on the platform, or uploading their own applications to the platform. The service is interfaced via a web browser or API, although the API only lets you run a preexistent workflow.

Din[11] acknowledges that this type of platform has at least the potential to alleviate the task of large scale, unsupervised gait analysis experiments of subjects in their home environment, but goes on to highlight a critical problem: in order for the service to be useful it must be able to run what is typically algorithms too sophisticated in terms of libraries and dependencies to be easily deployed.

Moving away from the idea of a general purpose research platform the work of Pan et al[6] introduces PD Dr. This work narrows the focus and concentrates on patients suffering from Parkinson’s disease. Realizing the value in monitoring subjects or patients in their daily life, the work of Pan presents a platform for gait assessment that allows the patient to conduct tests at home using an app which they call PD Dr. The app presents the patient with a set of tests and instructions on how to go about in performing them. The app uses the embedded accelerometer of the phone to collect data. After a test is completed, the data is sent to a cloud server for processing. The resulting output is sent back to the client and can also be accessed by a doctor via a database. The cloud component of PD Dr is modular in design and is argued to scalable. But according to the authors, the system is not designed to perform long term continuous monitoring of subjects.

The approaches on the literature indicate that to get the better results from gait analysis, subjects need to be monitored in their natural environment and ideally also during longer periods of time. Understanding that studies on large scale becomes resource intensive due to the amount of manual labor needed there is demand for a platform to overcome these problems.

Making use of personal devices such as smartphones helps in further lowering of thresholds for large scale gait analysis, this is demonstrated with the mobile application and cloud service PD Dr. While PD Dr allows for ‘at home’ gait assessment it lacks the general purpose of the e-Science central and long term monitoring capabilities.

We see that in all of the above examples there is no real mention of optimization of delivering output with short delay. The idea of getting results back in real time appears yet unexplored even though there might be applications of gait analysis which would benefit greatly from real time feedback. Nor are we presented with a solution for running *existing* algorithms in a cloud environment. The use of sensors embedded in personal devices such as smartphones is tried but only in a restricted setting, could the smartphone serve as a more general input device for gait analysis? Can a platform be designed as to support a typical gait analysis algorithm and also aimed at fast delivery of results?

3.2 Problem specification

One part of the platform will be the mobile application which will act as the collector of accelerometer data for the server. The collecting part will use the embedded accelerometer in the mobile phone. The server requires the accelerometer data to be delivered in a certain rate for the algorithm at the server to be able to work as expected. The phone will both act as sensor and as a gateway for external shimmer sensors. The user will in the interface be able to see the quality of the events in “real-time”, and that the gait analysis were ok.

The mobile application will have to handle accelerometer collection, sending the result to the server, saving the collected data in a database, plot the result from server and be able to do this simultaneously independent if the screen is on or off.

3.3 Approach

The development will take a bottom up approach. Beginning by developing a MVP (minimum viable product) and improving from there. The project will work in sprints according to the agile model. After each iteration the quality of the platform will be measured. The assessment of its quality will be based on the key features mentioned in *purpose.*

A standard test for evaluating each version must be developed. The standard test must be developed in a way so it can be an objective measure of the progress made along each critical axis (scalability, reliability, performance, accessibility and presentation). The test after each version will be helping in answering the question on what to do next.

3.4 Result verification

The result verification will be focused around reliability, performance and accessibility. There are several result verifications that needs to be made.

End-to-end response time: Measure the time it takes to send data from the phone and to get a verification back from the server in ms by logging.

Resource usage: CPU, memory, storage and battery: This will be measured directly in android studio and the application.

Network failure: how does the application handle the data when network/server goes offline. How much data can be stored locally before it need to start sending it to the server. This will be achieved by logging data and make a test in android studio.

App crashing: Make sure the code is written in a good way to minimise bugs and the chance that the app crashes by testing the code in android studio.

3.4.1 Reliability

The reliability in the mobile application will be measured by

- Resource usage
- Network failure
- App Crashing

3.4.2 Performance

The performance in the mobile application will be measured by

- Resource usage
- End to end response time

3.4.3 Accessibility

The accessibility in the mobile application will be measured by

- App crashing
- Network failure

3.5 UtExpo

At UtExpo a functional platform will be demonstrated with the ability to show the real time aspect as well as the detection of gait cycles with the mobile application as the collecting part and the cloud server as the processing part.

Timeplan

The development will take form using Scrum, which is an agile development method. It is organized in sprints, which can last between 2 and 4 weeks. For this project, each sprint will take 1 or 2 week(s).
First goal is to create an MVP (Minimal Viable Product) due to 2 february. During development and sprints, there will be time dedicated to documentation and writing on the report.

Phases	**Duration**	**Date**
Sprint 1 - MVP	2 weeks	16 jan - 30 jan
Project plan seminarium	-	3 feb
Sprint 2 - Improvements of MVP	2 weeks	6 feb - 20 feb
Sprint 3 - Quality iteration	2 weeks	20 feb - 6 mar
Sprint 4	2 weeks	6 mar - 20 mar
Midterm seminar	-	16 - 17 mar
Sprint 5 - Last coding sprint	2 weeks	20 mar - 3 apr
Writing report	ongoing	3 feb - 12 may
Final seminar	-	18 - 19 may
Individual examination	-	24 may
Utexpo, presentation of project	-	24 may
Last day for correlated report	-	4 jun

References

[1] *R. Williamson and B. Andrews, “Gait event detection for FES using accelerometers and supervised machine learning,” Rehab. Eng., IEEE Trans. on, vol. 8, no. 3, pp. 312–319, 2000.*

[2] *A. Salarian, H. Russmann, F. Vingerhoets, C. Dehollaini, Y. Blanc, P. Burkhard, and K. Aminian, “Gait assessment in parkinson’s disease: toward an ambulatory system for long-term monitoring,” Biomed. Eng., IEEE Trans. on, vol. 51, no. 8, pp. 1434–1443, 2004.*

[3] Loudon J, et al. The clinical orthopedic assessment guide. 2nd ed. Kansas: Human Kinetics, 2008. p.395-408.

[4] *D. A. Bruening and S. T. Ridge, “Automated event detection algorithms in pathological gait,” Gait Posture, vol. 39, no. 1, pp. 472–477, 2014.*

[5] *M. Yuwono, S. W. Su, Y. Guo, B. D. Moulton, and H. T. Nguyen, “Unsupervised nonparametric method for gait analysis using a waist- worn inertial sensor,” App. Soft Comp., vol. 14, Part A, pp. 72–80, 2014.*

[6] Pan, Rohit Dhall, Abraham Lieberman, Diana B Petitti, “A Mobile Cloud-Based Parkinson’s Disease Assessment System for Home-Based Monitoring,” [Online]. Available: 10.2196/mhealth.3956

[7] J. Rueterbories, E. G. Spaich, B. Larsen, and O. K. Andersen, “Methods for gait event detection and analysis in ambulatory systems,” *Med. Eng. & Phys.*, vol. 32, no. 6, pp. 545–552, 2010.

[8] *P. Missier, S. Woodman, H. Hiden, and P. Watson, "Provenance and data differencing for workflow reproducibility analysis," Concurrency and Computation: Practice & Experience, 2012.*

[9] *O.J Woodman, “An introduction to inertial navigation,”[Online]. Available: https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-696.pdf*

[10] S.Del Din, A.Godfrey, B.Galna, S.Lord and L.Rochester, “Free-living gait characteristics in ageing and Parkinson’s disease: impact of environment and ambulatory bout length,” Del Din et al. Journal of NeuroEngineering and Rehabilitation, 2016

[11] S. Del Din, A.Hickey, S.Woodman, H.Hiden, R.Morris, P.Watson, K.Nazarpour,M.Catt, L.Rochester, A.Godfrey, “Accelerometer-based gait assessment: Pragmatic deployment on an international scale,” 2016

[12] A.Weiss, S.Sharifi, MS, M.Plotnik, PhD, J.P. P. van Vugt, MD, PhD,N.Giladi, MD, and J. M. Hausdorff, PhD, “Toward Automated, At-Home Assessment of Mobility Among Patients With Parkinson Disease, Using a Body-Worn Accelerometer,” 2016

[13] H.Hiden,S.Woodman,P.Watson and J.Cala, “Developing cloud applications using the e-Science Central platform,” 2016

9 References

[1] Janice Loudon, Marcie Swift, and Alexander Stephania Bell. *Clinical Orthopedic Assessment Guide - 2nd Edition*. Human Kinetics Publishers, 2008.

[2] R. Williamson and B. Andrews. Gait event detection for fes using accelerometers and supervised machine learning. *Rehab. Eng., IEEE Trans*, 8(3):312–319, 2000.

[3] Alvaro Muro-de-la Herran, Begonya Garcia-Zapirain, and Amaia Mendez-Zorrilla. Gait analysis methods: An overview of wearable and non-wearable systems, highlighting clinical applications. *Sensors*, 14(2):3362–3394, 2014. ISSN 1424-8220. doi: 10.3390/s140203362. URL `http://www.mdpi.com/1424-8220/14/2/3362`.

[4] Wickström-N. Khandelwal, S. Gait event detection in real-world environment for long-term applications: Incorporating domain knowledge into time-frequency analysis. *IEEE transactions on neural systems and rehabilitation engineering,*, 24 (12):1363–1372, 2016. doi: http://dx.doi.org/10.1109/TNSRE.2016.2536278. URL `http://hh.diva-portal.org/smash/get/diva2:909015/FULLTEXT01.pdf`.

[5] Oliver J. Woodman, C Oliver J. Woodman, and Oliver J. Woodman. An introduction to inertial navigation, 2007.

[6] Jan Rueterbories, Erika G. Spaich, Birgit Larsen, and Ole K. Andersen. Methods for gait event detection and analysis in ambulatory systems. *Medical Engineering and Physics*, 32(6):545–552, 2017/02/07 XXXX. ISSN 1350-4533. doi: 10.1016/j.medengphy.2010.03.007. URL `http://dx.doi.org/10.1016/j.medengphy.2010.03.007`.

[7] Hugo Hiden, Simon Woodman, Paul Watson, and Jacek Cala. Developing cloud applications using the e-science central platform. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 371 (1983), 2012. ISSN 1364-503X. doi: 10.1098/rsta.2012.0085. URL `http://rsta.royalsocietypublishing.org/content/371/1983/20120085`.

[8] Parkinson's disease information page. `https://www.ninds.nih.gov/Disorders/All-Disorders/Parkinsons-Disease-Information-Page`. Accessed: 2017-02-23.

[9] A. Salarian, H. Russmann, F. J. G. Vingerhoets, C. Dehollain, Y. Blanc, P. R. Burkhard, and K. Aminian. Gait assessment in parkinson's disease: toward an ambulatory system for long-term monitoring. *IEEE Transactions on Biomedical Engineering*, 51(8):1434–1443, Aug 2004. ISSN 0018-9294. doi: 10.1109/TBME.2004.827933.

[10] Toshiki Iso and Kenichi Yamazaki. Gait analyzer based on a cell phone with a single three-axis accelerometer. In *Proceedings of the 8th Conference on Human-computer Interaction with Mobile Devices and Services*, MobileHCI '06, pages 141–144, New York, NY, USA, 2006. ACM. ISBN 1-59593-390-5. doi: 10.1145/1152215.1152244. URL `http://doi.acm.org/10.1145/1152215.1152244`.

[11] S. Del Din, A. Hickey, S. Woodman, H. Hiden, R. Morris, P. Watson, K. Nazarpour, M. Catt, L. Rochester, and A. Godfrey. Accelerometer-based gait assessment: Pragmatic deployment on an international scale. In *2016 IEEE Statistical Signal Processing Workshop (SSP)*, pages 1–5, June 2016. doi: 10.1109/SSP.2016.7551794.

[12] Walter Maetzler and Lynn Rochester. Body-worn sensors—the brave new world of clinical measurement? *Movement Disorders*, 30(9):1203–1205, 2015. ISSN 1531-8257. doi: 10.1002/mds.26317. URL `http://dx.doi.org/10.1002/mds.26317`.

[13] Oresti Banos, Rafael Garcia, Juan A. Holgado-Terriza, Miguel Damas, Hector Pomares, Ignacio Rojas, Alejandro Saez, and Claudia Villalonga. *mHealthDroid: A Novel Framework for Agile Development of Mobile Health Applications*, pages 91–98. Springer International Publishing, Cham, 2014. ISBN 978-3-319-13105-4. doi: 10.1007/978-3-319-13105-4_14. URL `http://dx.doi.org/10.1007/978-3-319-13105-4_14`.

[14] Di Pan, Rohit Dhall, Abraham Lieberman, and B. Diana Petitti. A mobile cloud-based parkinson's disease assessment system for home-based monitoring. *JMIR mHealth uHealth*, 3(1):e29, Mar 2015. doi: 10.2196/mhealth.3956. URL `http://mhealth.jmir.org/2015/1/e29/`.

[15] Java - overview. `https://www.tutorialspoint.com/java/java_overview.htm`. Accessed: 2017-02-13.

[16] Java garbage collection basics. `http://www.oracle.com/webfolder/technetwork/tutorials/obe/java/gc01/index.html`. Accessed: 2017-02-13.

[17] Meet android studio. `https://developer.android.com/studio/intro/index.html`. Accessed: 2017-02-13.

[18] Using ddms. `https://developer.android.com/studio/profile/ddms.html`. Accessed: 2017-02-13.

[19] Api - application program interface. `http://www.webopedia.com/TERM/A/API.html`. Accessed: 2017-02-18.

[20] Weather api - introduction. `https://www.wunderground.com/weather/api/d/docs`. Accessed: 2017-02-18.

[21] Json: What it is, how it works, how to use it. `https://www.copterlabs.com/json-what-it-is-how-it-works-how-to-use-it/`. Accessed: 2017-05-07.

[22] Database. `https://www.ntchosting.com/encyclopedia/databases/database/`. Accessed: 2017-02-18.

[23] A simple explanation of 'the internet of things'. `https://www.forbes.com/sites/jacobmorgan/2014/05/13/simple-explanation-internet-things-that-anyone-can-understand/#64f4b5fd1d09`. Accessed: 2017-02-23.

[24] From 1982 coca-cola vending machine to latest trend: What the internet of things means for business. `http://realbusiness.co.uk/tech-and-innovation/2015/07/15/from-1982-coca-cola-vending-machine-to-latest-trend-what-the-internet-of-things-mean`
Accessed: 2017-03-07.

[25] What's the difference between iot and ubiquitous computing? `https://www.quora.com/Whats-the-difference-between-IoT-and-ubiquitous-computing`. Accessed: 2017-03-07.

[26] What is accelerometer and how does it work on smartphones. `http://www.techulator.com/resources/8930-How-does-smart-phone-accelerometer-work.aspx`. Accessed: 2017-02-28.

[27] Wearable sensor technology. `http://www.shimmersensing.com/`. Accessed: 2017-03-07.

[28] What are cloud servers? `http://www.interoute.com/what-are-cloud-servers`. Accessed: 2017-03-07.

[29] What is websockets? `https://kaazing.com/websocket/`. Accessed: 2017-02-28.

[30] What is cpu? `http://www.webopedia.com/TERM/C/CPU.html`. Accessed: 2017-06-02.

[31] Understanding user and kernel mode. `https://blog.codinghorror.com/understanding-user-and-kernel-mode/`. Accessed: 2017-03-27.

[32] Github. `https://github.com/`. Accessed: 2017-03-06.

[33] Trello. `https://trello.com/`. Accessed: 2017-03-06.

[34] Slack. `https://slack.com/`. Accessed: 2017-03-06.

[35] Sqlite. `https://www.sqlite.org/about.html`. Accessed: 2017-03-07.

[36] Websockets vs rest api understanding the difference. `https://www.pubnub.com/blog/2015-01-05-websockets-vs-rest-api-understanding-the-difference/`. Accessed: 2017-06-01.

[37] Android data storage. `https://developer.android.com/guide/topics/data/data-storage.html`. Accessed: 2017-06-01.

[38] Tcp over ip bandwidth overhead. `http://packetpushers.net/tcp-over-ip-bandwidth-overhead/`. Accessed: 2017-03-20.

[39] Data usage monitor. `https://play.google.com/store/apps/details?id=com.andcreate.app.trafficmonitor&hl=en`. Accessed: 2017-03-26.

[40] Decimation(signal processing). `https://en.wikipedia.org/wiki/Decimation_(signal_processing)`. Accessed: 2017-06-02.

[41] Cloud computing saves energy. `https://www.scientificamerican.com/article/cloud-computing-saves-energy/`. Accessed: 2017-06-02.

Tim Svensson

HALMSTAD
UNIVERSITY

PO Box 823, SE-301 18 Halmstad
Phone: +35 46 16 71 00
E-mail: registrator@hh.se
www.hh.se