Walking Aid Usage Prompt System

Pedro Caetano, Matthew Culley, Sean Coaker, Panayiotis Melios

Submitted to Swansea University in fulfilment
of the requirements for the Degree of Masters of Science
in collaboration with Bangor Health Clinic

Swansea University
Prifysgol Abertawe

Faculty of Science and Engineering
Swansea University

December 14, 2021

Contents

Chapter 1

Topic

This chapter details the problem and task our project aims to solve along with the limitations that could affect our project, including an analysis of the current solutions our project has to compete with. We will provide background research on dementia patients and senior citizens with similar conditions (hereby depicted as the 'User(s)') and their issues with forgetting their walking aids, and how wearable devices can have a varying psychological impact on them. Finally, we will detail our current progress consisting of two initial meetings with our client, and our takeaways from these engagements.

1.1 Background

The client came to us with the idea of developing a product for those suffering with dementia, a syndrome that is usually associated with a declining functionality of the brain. Dementia can manifest itself with a large range of symptoms, most commonly including memory loss, loss of mental sharpness, or loss of the use of language [**?**]. More importantly to our project, one other symptom can be a loss in movement skills, or an increased difficulty in moving. Dementia has been recognised as an illness that causes an increase in falls within patients [**?**]. As dementia affects mainly more elderly population, falls are often more dangerous to them. It is therefor not uncommon for dementia patients to be using walking aides to help mitigate this, a walking aid is only effective if it is used by the patient, and as explained by the client, this is something the users may struggle with remembering to use

1.2 The Problem

Upon completion of our initial meeting with the client, we clarified the motivation behind this project and the problem that we are working together to solve. The problem is to develop a solution that detects when a user is moving without their walking aid and reminds the patient (with a recorded message by a friend or a relative) to take their walking aid with them. Initial discussions

between ourselves and the client identified current issues with users feeling uncomfortable in being forced to wear foreign objects, meaning we would need to take this into account when developing our solution. We also clarified that users get easily alarmed and frightened by generic "obnoxious" alarms, often associating them with those notifying them of danger, such as fire alarms. The client suggested that we facilitate a recording feature within our solution that would allow recognisable voices to the user to remind them to use their walking aid.

1.2.1 Similar Solutions

Current solutions include the use of locator systems that allow users to easily track down valuables such as keys or a wallet. Such systems include the Tile ecosystem which allows a user to attach a Tile device to their valuables and then use a smartphone app to trigger an alarm from the Tile device that notifies the patient of the location of their valuable. As previously mentioned, users can get frightened and disorientated by the sounds of alarms often associating them with danger rendering these forms of solutions unsuitable for our problem. This is without considering how difficult a users mind finds navigating through a smartphone device to open an application and request their Tile device to ring an alarm to help them identify the location of their valuable. Another problem with such approaches, is the lack of functionality for sending an alarm to the patient if they start walking without the aide. Solutions such as Tile would help a patient find the aide if it had been misplaced but implementing functionality to remind the patient to find the aide if they have already started walking without it, is lacking entirely. Other more legacy systems that carers may use to notify themselves that their patient is moving include hanging items from door frames that clatter together when the patient walks through the door or adding pressure pads under door mats that sound an alarm when the patient steps on the door mat. Our solution is focussing on the protection of users that are alone and wanting to move around their home or ward, meaning that door mat pressure pad based solutions and methods for alarming a carer would be insufficient. We also would like our design to be more elegant than those rudimentary solutions, if the patient is walking with the aide, they will still get an alarm from the pressure pad. A well implemented system could alleviate both those problems, and increase the quality of life of the patient and facilitate the carer.

1.2.2 Limitations

Our main limitation for our project is that we need to develop a discrete device that will not make the users feel uncomfortable in any way and will minimise discomfort to an extent where it is acceptable for day-to-day life. Early plans for the device lean towards a watch style device that the patients can wear on their wrist to track their movement. If we were to create a wrist wearable device then we would need to ensure that the footprint of device is small enough to be worn comfortably. This limits the hardware that we can feasibly use for our project. We also need

to consider the hardware being used and how they can all be fit within a wrist device. The head of Swansea University's Embedded Systems module has kindly offered to supply us with ESP32 based TinyPICO devices which would be suitable for this project due to their small form factor. We would need to consider how an accelerometer could be attached to the TinyPICO to allow both devices to fit within a watch casing. If we could find hardware small enough to build into a watch-based prototype, we may still run into issues with the patients wanting to use the product. During our meeting we were told that patients already had to wear wrist tags or similar items. Adding more items the patient needs to wear is unlikely to be well received. Our form factor could take the shape of something that clips onto what the patient is already wearing, such as clothes or belts, or a tag that they already use.

Other limitations for the wrist device are that it should not contain any strong LEDs, obnoxious vibration motors or alarms to avoid startling the user. Avoiding the use of LEDs would be of benefit to us here as it would allow the watch device to save battery during operation. Another limitation to the watch device is that it would need to be power efficient and avoid the users needing to frequently charge the device. The patient needing to do this would work against our goal of creating a user-friendly experience for them. The ESP32 chips included on the TinyPICO boards utilise a facility called 'deep sleep' or hibernation states, which effectively powers down certain modules connected to the board. We could create a system here that fires an interrupt when the accelerometer detects movement, then forcing the ESP32 to wake up and handle the interrupt. The device could therefore be in sleep whilst the patient is static, to save battery.

The device attached to the walking aid has much less limitations, and so we do not need to consider such a small form factor as it is not being worn but will still be using a TinyPICO to encourage compatibility and interchangeable/modular development experiences. Our limitation with this device is to also disable the use of LEDs to avoid startling the patients, and to include a speaker and microphone to allow a relative or carer to record a voice note which will be played to remind the patient to take their walking aid with them when moving. The TinyPICO boards include a very minimal amount of storage space and so we may need to include a SD card to store the recordings. We will also be limited to the budget of £150 that we have been assigned and must ensure that all the devices needed to build the system can be purchased within our budget.

1.3 Current Work

As stated earlier in this chapter, we have held initial meetings with the client where we have clarified the problem they aim to solve with this project and outlined the project scope. On the 18th of November we held an introductory meeting with the client where we gained an understanding of what the problem is and what kind of system the client was expecting to be produced. We clarified that we would need to gain our supply of hardware ourselves and that a budget of £150 would be allocated to us to aid with the procurement of the necessary hardware devices. However, within

this initial meeting we failed to identify the final direction the client wanted the project to head down and instead came away with the option of either developing a wearable device that would detect when the users was moving, or to use a non-wearable device such as a pressure blanket that would detect when the patient had got up from where they were static. We agreed with the client that we would schedule a second meeting for the 25th of November and within that time analyse the advantages and disadvantages to each method of developing the solution. We then agreed that we would return with a solution that we thought would best suit the design brief and that would best suit the development talent available to us within our team.

Within our own intra-team meeting we decided upon building and developing a wearable device solution over a non-wearable solution due to the extra features that could be included into a wearable device such as a fall detection system, a system recommended to be included by the client. We felt that despite a non-wearable device being a plausible route to take the project down, that factors such as a users moving off a pressure pad without actually standing up and walking would diminish the effectiveness of our solution.

On the 25th of November we hosted our second meeting with the client and established the team's preferred route for the development of this project. The client was content with this and agreed that the solution should be developed as a wearable device. We finalised the £150 budget with the client and agreed that our next steps would be to complete our milestone 1 document, including user requirements, and compiling a list of necessary hardware to develop the project. Our next meeting with the client is scheduled for December 16th where we will finalise the user requirements and compile a list of hardware to be purchased with the budget made available to us.

1.4 Project Aims

For evaluation purposes at the conclusion of this project, we will detail within this section a list of aims that should be met along with lower level objectives that define a set of criteria that will allow use to meet said aims. An evaluation within milestone 3 will look to compare the final product produced against the aims set out in this section in an attempt to gauge how successful our project is. A list of our aims and their lower level objectives can be seen below.

- Develop a solution that reminds the user(s) to take their walking aid with them when attempting to walk.
  - We should create a wearable device that includes a tri axial accelerometer to detect when the user(s) have started moving.
  - A device should be connected to the walking aid that also contains tri axial accelerometer that will be used to detect if the walking aid is moving whilst the wearable device is moving.

  - The walking aid device should contain a microphone and speaker that allows a carer to record a reminder for the user(s) in attempt to avoid startling the user(s), which could happen when using generic alarms.

- Create prototype devices that avoid startling the user(s) and avoids making them uncomfortable.
  - The wearable device should be designed and built such that it is inconspicuous in attempt to not draw attention to it from the user(s).
  - Unless the user(s) is deaf, we should avoid the use of LEDs, vibration motors, and the use of generic alarms.
  - The wearable device should be developed to be worn on the wrist rather than on a more uncomfortable body part such as the neck.
- Produce a solution that the client concludes is satisfactory.
  - Produce a document of user requirements and receive confirmation from the client that the requirements are sufficient.
  - Allow for changes in requirements during project development. Our chosen agile methodology will allow for easy integration of changed user requirements here.
  - Develop the solution such that it is compliant with the user requirements.

Chapter 2

Requirements

In this section, we will detail the project's functional and non-functional requirements, which are broken down into higher-level user requirements as well as lower-level specifications that will describe the process our team will go through to ensure the user requirements are met. These written requirements have been demonstrated to the client and they have provided their consent for the project to be based upon them.

2.1 Functional Requirements

Table 2.1: A table of functional requirements split into user requirements and their relevant specifications needed to meet those user requirements.

Code	User Requirement	Specification
FREQ1	The wearable device should detect when a patient has walked more than 1 metre before communicating with the walking aid.	We can use a tri axial accelerometer to detect changes in acceleration that are indicative of the user moving or being mobile. Once movement is confirmed, we will then commuicate with the walking aid device to ensure the user has successively reached and engaged with it, prior to alerting them to use it.
FREQ2	Patients should be alerted with the voice of a friend, carer or relative to avoid startling them.	The device to be attached to the walking aid should include a microphone and speaker that will allow the user to record a voice note and store it on the device. We may need to include an SD card within this device that will store the voice note if need be.

Continued on next page

Code	User Requirement	Specification
FREQ3	The wearable device should include a solution for deaf people that still reminds them to take their walking aid with them without the need for an audio alarm.	The wearable device could use a vibration motor here that vibrates to remind the user to use their walking aid. We can also utilise the LEDs on board the TinyPICOs to flash to remind the user also. There are issues here with potentially startling the patient with the use of vibration and LEDs, however we feel this is most feasible method for meeting this user requirement.
FREQ4	If development time allows, the system should include fall detection as a stretch goal feature.	Using the tri axial accelerometer mentioned in the specification of FREQ1, we could detect acceleration and movement along the negative side of the y-axis in attempt to detect when the patient has fallen. An alert system can be used in accordance to alert a nearby carer or relative.
FREQ5	The wearable device should communicate to the walking aid device to let it know when it's started moving.	To meet this requirement we investigate the use of 433MHz Rx/Tx modules for low power and low level communication between the 2 devices in the system, this technology should allow for the basic level of communication required, with minimal power use and minimal complexity.

2.2 Non-Functional Requirements

Table 2.2: A table of non-functional requirements split into user requirements and their relevant specifications needed to meet those user requirements.

Code	User Requirement	Specification
NONFREQ1	The watch should be a small enough form factor to fit on the wrist of the patient.	Deciding to use TinyPICO devices as the main board of the device will allow us to keep the device to a small form factor given the TinyPICO is 18mm x 32mm. We will also take into account the form factor when deciding upon extra hardware to add to the devices.
NONFREQ2	The devices shall be power efficient to avoid the patient needing to charge them often.	The TinyPICO devices we will use as the main boards for the devices include an ESP32 chip capable of using deep sleep cycles. These cycles allow the ESP32 to power down non critical components in order to save power. We can create an interrupt within the code here that powers the devices on when an alarm needs to be fired due to the patient moving. This means that the devices will only need to be fully powered on when movement is detected.
NONFREQ3	The devices shall avoid startling the patients with the use of LEDs and vibrations unless they are deaf.	In this case we would power down the LEDs and Vibration motor at all times to avoid startling the patient. Powering down these devices will also allow us to save battery.
NONFREQ4	The wearable device should be discrete enough that it does not make patients uncomfortable wearing it.	We intend to design the device to make it as close the design of a watch as possible, keeping it small and sleek so that it looks like a fashion accessory rather than a medical device. Using small hardware devices we can keep a small form factor so that the device is not overly noticeable on the patient's wrist.

Continued on next page

Code	User Requirement	Specification
NONFREQ5	Security of devices should prohibit outside devices from communicating with the network.	A possibility here is using an agreed upon 'sync word' between our 2 devices that only reads communcations from devices using the same 'sync word'. This would stop other devices being able to communicate with the network unless they knew the sync word being used.
NONFREQ6	The developed prototype wearable device shall be designed such that it can fit various wrist dimensions.	The strap of the wearable device should be designed such that it can be adjusted in a similar way to how normal watches can be adjusted.
NONFREQ7	The coding system the devices run on should be efficient enough to react to real time actions.	Our code should be developed with efficiency in mind that will allow for real time readings from the accelerometer to precisely detect when movement has passed 1 metre in order to notify the walking aid device that this has happened. The Rx/Tx communication module will allow for fast communication between the devices to ensure that user is promptly reminded to take their walking aid with them.
NONFREQ8	The system should be delivered upon its conclusion with relevant documentation, including a user manual.	Our milestone 3 document will include the necessary documentation needed to be handed over to the client upon the handover of our project. This milestone will include a user manual along with any designs and testing documentation that were created during the development of the product.

Chapter 3

Specification

3.1 Description

This project includes the development of a wearable device that is able to detect when the user is/has moved, and a remote device attached to their walking aid, that is able to remind them to use it if they have not done so. A prerecorded message from a family member or relative will act as the notification. Both devices have to be discrete and unobtrusive, as the users are known to be easily startled by new and foreign devices near them, as well as with lights/vibration and other stimulus easily confusing them. The message will be stored locally, either on the hardware products inbuilt storage, or an external SD card. In the eventuality that the user is deaf or hard of hearing, the device will also have to have support for alternatives stimuli, such as vibration or light based feedback. Within this section we will provide a brief specification of the project along with references to requirements specified in chapter 2.

3.2 Software Behaviour

The wearable device must be able to communicate with the walking aid device for the project to be successful (FREQ5). communication should occur once the wearable device has detected that the user has walked more than 1 metre (FREQ1). Then, the wearable device should communicate with the walking aid device to check if the walking aid is moving also. If it's not, then the walking aid device will play a pre-recorded message from a carer, friend or relative reminding the user to take their walking aid with them (FREQ2). Once the walking aid begins movement then the reminder will stop being played.

Furthermore, dementia effects all types of people at varying ages. That means that the wearable device must be able to fit on anyone's wrist and must contain hardware small enough to allow the wearable to fit comfortably on the wrist (NONFREQ1). Because we do not want patients to feel uncomfortable while using the wearable device, it will be designed to look like a watch. Apart from the device's appearance and dimensions, the device should be power efficient, as the patients may forget to charge it (NONFREQ2). Aside from the device's appearance and dimensions, the wearable device's material must be durable in case it is dropped. That is why the material of the wearable device will be ABS plastic, which is an opaque thermoplastic and amorphous polymer and easily printed with our printer.

Should time allow, the device will also include fall detection functionality as part of an agreed stretch goal (FREQ4). A tri-axial accelerometer will be used to determine the user's hand height, and more partic-

ularly the device's height. If the user's hand is extremely close to the ground, it is likely that the user has fallen, and if the device detects that the user has been on the ground for an extended amount of time it will begin an emergency procedure to notify an in case of emergency contact.

In terms of the security of the system, the wearable device and the walking aid device will communicate only with each other in pairs. This can be achieved by using a unique synchronization mechanism between each device. As a result, an external device will be unable to communicate with the walking aid or even the wearable device within the network (NONFREQ5).

3.3 Performance

Our main concern with the performance of the device is to ensure that the battery usage is kept at a minimum level (NONFREQ2). We have previously discussed how deep sleep cycles will be used to do this and is necessary to avoid issues where the user may forget to charge the device rendering the device useless. Other performance concerns centre around the communication between our 2 devices, where the process should use a low amount of power and should allow for consistent and robust communication (FREQ5). Rx/Tx modules will provide the basis to meet this performance criteria. Finally, the execution of processes by our device should be instantaneous to allow the device to detect and respond to real time movements as quickly as possibleU̇sing a stripped back language such as Arduino C or Micropython will allow for this and will allow the developers the inclusion of unnecessary overheads within the code that is uploaded to our TinyPICO devices.

Chapter 4

Methodology

4.1 Software Developement Methodologies

4.1.1 Development Methodology

Due to the nature of our team and project, We decided to implement an agile based development methodology. As a small team, it is easy for us to communicate with each other about the project, which allows us to be aware of what the other members of our team are doing, both of which are key factors for choosing an agile methodology. The question There are a variety of methodologies available, each with their own upsides and downsides, which are discussion below. Based on our research into these, we settled and agreed to use a scrum-based development methodology. Scrum is a combination of iterative and incremental development. Allowing for a best of both worlds approach of having early builds working, but also being agile, and able to add requirements during the development process [**?**].

The scrum methodology is rather simple. To work effectively, it requires collaboration between an appointed 'Scrum Master' and the rest of the development team, as well as a 'Product Owner'. The members will work in close collaboration with each other, and have multiple, continuous iterations of the software builds to create a finished product. The role of the scrum master is to eliminate impediments [**?**], and to create the conditions that allow the development team to work in their most effective manner, they take a leading role in choosing what sprints the team work on and are crucial when team splitting decisions are required.

A typical scrum workflow consists of iterative scrum cycles, typically lasting one to three weeks each. Initially a requirements backlog is created. This is done in collaboration with the client and is the documentation that describes all the requirements that the software must meet. Essentially, it is a description of the product that the team are aiming to build. After the initial backlog is established, the scrum cycles can start. As before, each scrum cycle lasts typically up to three weeks. The cycle begins with the team, headed by the scrum master deciding which requirements to prioritise and implement in the upcoming cycle. The aim is to have a potentially shippable/working product at the end of each sprint, although this may not be possible immediately due to the workload required to build a framework, following sprints are expected to get increasingly more successful. Once the team have allocated the suitable requirements and resources to the sprint, the sprint begins. The team works on the project, meeting each day in what are known as 'scrums' to review progress made on each day and discuss any problems or insight they have found on the

implementation. The team will also meet and hold a sprint review, with the product owner. This demonstrates the product and how it has been developed and the progress since the last sprint. The team will also review the previous sprint, to see if any changes need to be made in anticipation of the next one. [**?**]

For our project, we will be slightly modifying the implementation of scrum. The above-mentioned workflow is designed as a framework for full time software developers, which we are not. As students with other commitments, we will not be able to meet daily to discuss the project or work on it fulltime. It would be an unrealistic target to hold daily scrums, but we can still follow our adaptation of the methodology. We will aim to hold scrums at frequent intervals, most likely every week, other coursework dependant. This wil allow us to keep track of current sprints, as well as compensate for this by being realistic in what we can achieve in the given sprint timeframe, and what we set as the sprint's goals.

Ultimately, we chose to use scrum for a variety of reasons. Primarily because we realised that as a small team, frequent communication would be easy to achieve, so we would be able to make use of this framework in the most effective manner, compared to other methodologies. We would be able to meet frequently and use that to ensure progress is made on the project. Scrum works well with this, which enables us to work together to frequently get working, testable products and allow us to implement more features that would otherwise rely on this. We also feel that using scrum would allow us to have a higher quality product, as the frequent iterations during sprints would allow us to streamline our implementation and constantly improve it, which also minimises risk.

In comparison, the waterfall methodology where implementation begin much later in the lifecycle once all documentation and test suites are formulated. Using scrum, we start programming earlier, giving us more time to deal with any difficulties we encounter. It allows us to identify problems earlier, have longer to address them, or encourage us to plan alternative implementations that avoid such development deadlocks.

We chose scrum over other pure agile methodologies mainly because we thought it was the best fit for our team and situation. Other methodologies, like extreme programming, could have worked well, but didn't suit our situation as students as well and added unnecessary risk to an already time critical project. For example, extreme programming requires pair programming, something that would be hard for us to do consistently and would drastically diminish our overall productivity.

4.1.2 Hardware

We have chosen to base our hardware on the TinyPico implementation of the Espressif Systems ESP32 family of devices. This is mainly due to our experience with the platform, including previous university modules experimenting with these microcontrollers. To compliment the TinyPico, we will need to include an array of input sensors, as well as ways to output information to the user. In order to detect movement, we will rely on accelerometer data from the pervasive ADXL family of accelerometers, namely the ADXL345 due to its reliable and proven performance, as well as its easy availability and team members' experience with the device. Beyond what's available on the TinyPico, we may also require additional methods of receiving and transiting data to and from our wearable device and the walking aid device, we have a few technologies we are considering, such as 433MHz Rx/Tx, or the proprietary LoRa communication system if the budget allows for it.

For outputs, the wearable device will need an output LED that may be handled by the TinyPico itself, or most likely an external low voltage LED to allow for more flexible placement, furthermore, it will need support for vibration feedback, which will likely come in the format of a mobile phone style vibration

motor. This will ensure the users are not startled, due to its lower power and smaller size The walking aid device will require to have a small speaker as an output, in order to prompt the user to use their walking aid. We can then either include a microphone onboard the device to record and store data to the onboard TinyPico storage, or support the use of an external SD card to access a prerecorded message.

One of the group members has access to a 3D Printer and has experience in 3D Design, and such we will be able to rapidly develop and test chassis and shells for our sensors, to more close resemble a finished proud whilst testing or demoing to our client. This is crucial due to the importance of the form factor and readability of the devices, as previously discussed in the document, such as minimising patient discomfort.

The TinyPico form factor is incredibly compact, and would be viable for the NONFREQ1 requirement. This also extends to the input and output sensors, which we also believe would easily fit on a wearable or walking aid based device. The low power consumption of the TinyPico and sensors, will help us ensure NONFREQ2 and NONFREQ3 are met. This can be further exploited by implementing deep sleep algorithms within the TinyPico's routine, further reducing power consumption thus extending battery life on a single charge.

4.1.3 Software

Whilst working on the project, our team will use a plethora of services to assist in communication and management of the project. We have decided to use the Discord platform to keep in touch and communicate. This allows us to organise our channels and information streams accordingly, and even has essential tools for peer-development such as voice/video calls, with screen sharing support. We started a git repository on GitHub, which is where our source code and work towards our project deliverables will be. This version control software allows us all to work on the tasks together or independently, and for progress to be easily managed, tracked and merged with everyone else's work.

Our milestones and written deliverables will be written in Latex, and stored on the GitHub to allow each team member to use the IDE or editing environment they are most familiar with, this ranges from Overleaf, to Intellij, to less specific editors like ATOM or SublimeText

The TinyPico supports Arduino C, which is a subset of C with an essentially very similar, but tailored featureset and commandset. We have some experience with this in the past, however we will keep an open mind towards Micropython, which promises similar levels of performance and flexibility, in an arguably easier to learn and use language which we have more experience with.

4.2 Testing Strategy

To perform successful and consistent testing across the development of our project, we have decided upon a testing strategy that our team will follow. We have designed the testing strategy around our small team, following the schedule detailed earlier in the document, and complies with our chosen software lifecycle methodology of an adapted scrum methodology. Our testing strategy consists of mainly integration testing, which we will perform at the conclusion of the scrum sprints. A sprint consists of a block of time where our developers will develop code for a specific feature that needs to be added to the already built system. Integration testing will be performed at the conclusion of these sprints to ensure that the features being added to the system do not create any additional issues within the system, and that its role in the system is performed as expected.

Additionally, the features being added at the conclusion of a sprint need to be tested throughout the sprint periods. The testing of embedded systems can be a very manual process due to the fact that many of these systems rely upon actions from the outside world to effect how the system responds. We will still aim to include unit testing when developing our features to ensure they are working correctly before they are implemented into the larger system, possibly using expected real world values in place of direct sensor data. Such unit tests could consist of checking that correct outputs are produced when certain inputs are passed into a function. For example, an input of distance walked could be passed into a function along with a boolean value that states whether the walking aid is moving or not. We would expect here that a notification is output to the walking aid when the distance walked is greater than 1m, but no notification is otherwise expected. Therefore, we could unit test this to ensure that the correct message is output when a certain criteria of input is passed into the function. The unit tests themselves could be classified as white box testing and will be used to test the inner workings of the system in an attempt to identify bugs that are hidden within the implementations of functions or through the use of complex code. However, we will also apply black box testing to our features to test the more broad functionality of the component being worked on, within the scope of the system

As previously mentioned, embedded systems involve hands on and manual testing within a real world scenario. Manual tests will allow for the effects of real life events to be simulated, and thus the product tested during production. Such events could consist of testing the fall detection feature (which is a stretch goal, but is a good example) which cannot simply be tested within the code itself. We would instead simulate the inputs of the device falling to the ground and then checking if an emergency message has been fired by our wearable device. Manual tests on our embedded systems will be documented and will include the procedures taken to run the tests and the effects this had on the system. Our testing will then be evaluated to ensure that the behaviour of the system matches the specification and the expected outputs.

Chapter 5

Project Management

5.1 Team Roles

Within this section we will detail roles needed to be fulfilled within our team that aligns with the context of our project and our chosen software lifecycle methodology. Each role will be assigned team members that are responsible for the tasks being undertaken by said role. Due to the nature of our scrum methodology, we will be avoiding the use of a designated team leader and instead opting for a self-managing team [**?**]. A list of our team roles can be seen below, along with their assignees and detail about what each role entails.

5.1.1 Scrum Master

Assignee: Sean Coaker
Description: The scrum master within our project will not act as a team leader, but will instead facilitate and manage the process of the scrum software lifecycle methodology [**?**]. Due to the adjustments we made to our scrum methodology, such as avoiding daily meetings due to university commitments, our scrum master will need to ensure that our team adheres to our adjusted methodology and time plan set out within this document. Therefore, our scrum master will need a clear understanding of the team's selected software lifecycle methodology and should regularly reference it to provide consistent guidance to the team when developing the project.

5.1.2 Full Stack Developer

Assignee: All Team Members (Sean Coaker, Pedro Caetano, Matthew Culley, Panayiotis Melios)
Description: Our full stack developers will handle the code development for all aspects of the project. They will develop code for both the walking aid and wearable devices, implementing code that allows our project to meet the functional and non-functional requirements detailed within this document. The developers will need to quickly adjust to the new software and hardware they encounter during the development of this project to ensure that the final product is developed on time. As mentioned previously, our team is self-managed and therefore regular communication will need to occur between developers to keep everyone up to date with progress being made.

5.1.3 Communications Officer

Assignee: Sean Coaker
Description: To provide communication between the development team and the client, we elected a communications officer. The communcations officer shall be the main point of contact with the client, dealing with queries, organising meetings and ensuring that the client requirements are communicated correctly to the development team. The communcations officer has so far organised initial meetings with the client, and submitted documented user requirements to the client, which the client confirmed they are content with.

5.1.4 Prototype Design Officer

Assignee: Pedro Caetano
Description: Due to the nature of our project, we needed to assign a prototype design officer to handle the design and development process of the device casing. We elected Pedro for this due to his previous experience in working with 3D printers and feel that his experience can transfer well into developing the designs for our device casing, along with being able to transfer these designs into 3D printed useable prototypes.

5.1.5 Testers including Quality Assurance responsibilities

Assignee: All Team Members (Sean Coaker, Pedro Caetano, Matthew Culley, Panayiotis Melios)
Description: As mentioned previously, our team is self-managed meaning we wanted to avoid the use of lead roles such as team leaders, testing leaders and quality assurance leaders. For this role, all team members will be expected to perform testing procedures that align with our testing strategy. Along with this, each team member will also take on quality assurance responsibilities to ensure that our product is of the highest quality the team can possibly produce. Our testing role also includes the responsibilities of documenting test results to allow the team to collate them at the development of our milestone 3 document for proof of the success of our product.

5.1.6 Documentation Developers

Assignee: All Team Members (Sean Coaker, Pedro Caetano, Matthew Culley, Panayiotis Melios)
Description: As we are a small team, it is a requirement that all team members contribute to the development of the documentation required to be completed within this project. The fact that all members assigned to the development of the documentation are also full stack developers can be beneficial here as they can provide an in-depth understanding of the developed system when detailing it within our documentation.

5.2 Time Plan

5.2.1 Gantt Chart

We built a Gantt Chart for the project schedule to allow us to keep track of the time. We couldn't create tasks that would indicate a slippage since it would disrupt the critical path. As a result, we increased the time alloted to each assignment, reflecting the slippage. We included slippage due of other university homework, exams and even illness.

As shown in Figure **??**, the highlighted tasks illustrate the project's critical path. We devided the project into three Sprints because we are utilizing the scrum methodlogy. The first Sprint consists only of require collection and project planning. The second Sprint includes the sprint plan, as well as the software and hadrware designs and implementations. The third Sprint contains the sprint plan as well as the system's final implementation, in which the walking aid will connect with our system, the wrist device. The final task of each Sprint is a milestone, signaling the Sprint's completion.

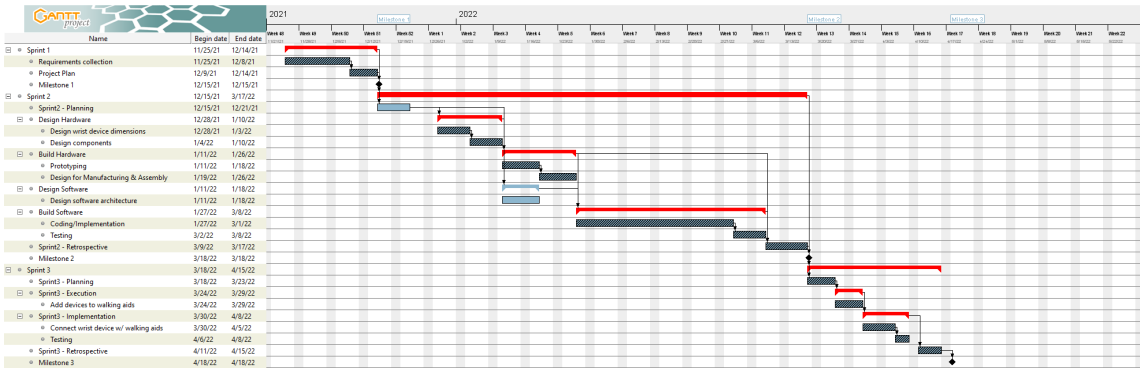

Figure 5.1: Gantt Chart

5.2.2 Activity Network Diagram

In terms of scheduling, an activity diagram is just a significant as designing a Gantt Chart. We merged several of the jobs that were introduced to Gantt Chart into one task to make the activity network diagram more undestandable. However, this has no bearing on the total time requried to finish the project. According to Figure **??**, the total time requried is 107 working days.

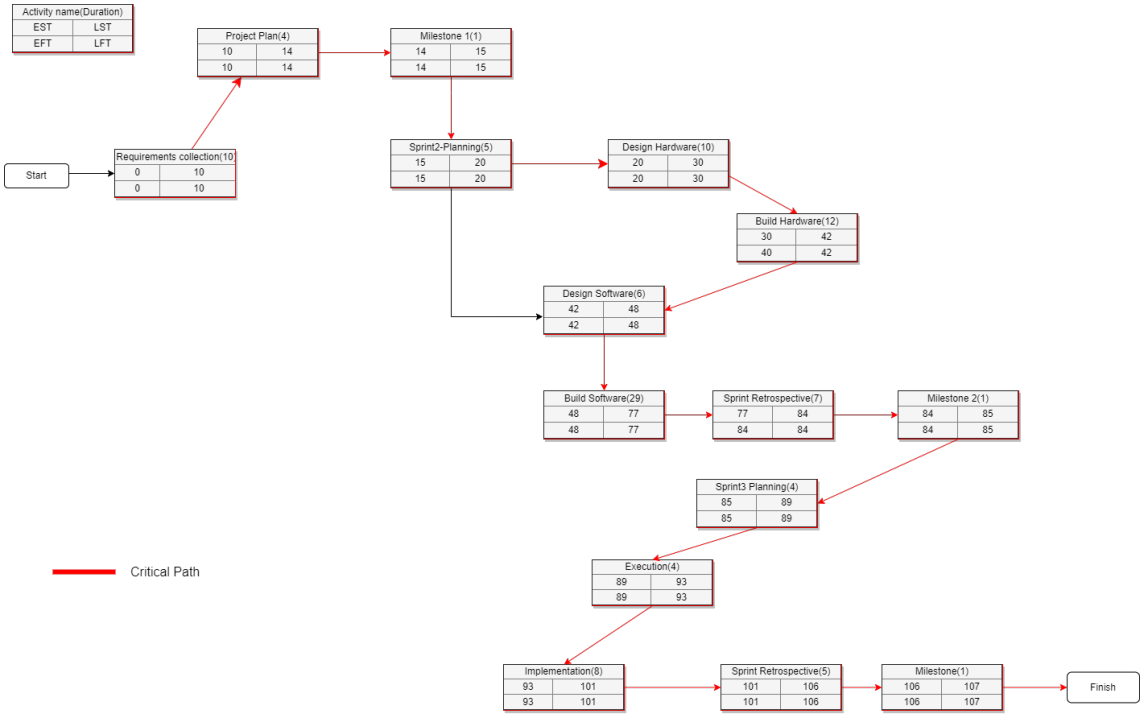

Figure 5.2: Activity Network Diagram

Chapter 6

Risk Analysis

Within this section, we have included a risk analysis upon the proposed project by providing a list of possible risks that could negatively affect it. We will provide a list of these risks along with mitigation strategies that we will use to avoid the occurrence of said risks. Upon completion of the risk analysis, we will include a matrix demonstrating the likelihood of a risk occurring and the impact that risk could have on the project should it occur.

6.1 Risk Identification and Mitigation Strategies

Table 6.1: A table of risks along with strategies to mitigate those risks.

Code	Risk	Mitigation
RSK1	Our hardware devices may fail and will limit development and testing.	To mitigate this risk we will choose to use low cost but still effective hardware, which will allow for extra funds within our £150 budget should we need it to replace hardware during development. We also have the opportunity to attain TinyPICO devices from the University for this project, allowing us to minimise the effects on our budget.
RSK2	The Rx/Tx modules could fail disabling communication between the wearable and walking aid devices.	The replacement of these modules should not be much of an issue due to their low cost. The real issue would arise when an Rx/Tx module fails whilst in operation for a patient. We would need to form a protocol here that detects when communication is unable to occur between the 2 devices, and can alert the patient's carer of this.

Continued on next page

Code	Risk	Mitigation
RSK3	Uploading erroneous code to our TinyPICO devices could brick the TinyPICO devices.	Should this occur, we could attempt to reflash previosuly working code onto the TinyPICO. If this fails, we would be left with the occurrence of risk RSK1 where we would need to replace the TinyPICO devices that have been bricked. The low cost of the TinyPICO devices should enable us to purchase some replacements if need be.
RSK4	GitHub experience a malicious attack or a server failure which could cause our repository to be lost.	Mitigating this risk is difficult. It's unlikely this will happen and that we would lose our repository as GitHub likely uses a vast backup storage solution. But, should it happen it would be catastrophic and so we should mitigate against this risk. To do this, each developer within the team will store a clone of the repository on their personal system and the team will be able to piece the code back together should this risk arise.
RSK5	The user requirements we accept could be too large to implement within the given timeframe for the project, potentially leaving the client disappointed at project handover.	To mitigate against this risk, we have discussed our user requirements with the clients and feel that we have decided upon a set of features that we feel we can confidently implement within the time frame given to us. We have also some marked some features as stretch goals to ensure that the most important features are added first.
RSK6	University commitments could impact the development and testing of the product.	We have attempted to account for this within our schedule by allowing for slippage. Slippage time could allow for the development of unfinished features. We could also spread unfinished work between developers as extra work in an attempt to complete the development of features on time.
RSK7	The hardware we choose to use may lack the libraries and compatibility with other hardware for quality feature development.	To avoid this, we will select hardware that is compatible with the Arduino ecosystem to ensure that they are compatible with each other and that libraries are available for code development.

Continued on next page

Code	Risk	Mitigation
RSK8	Natural disasters could bring about the loss of hardware and software being used for the development of our product.	The use of low cost hardware in this system will allow us to replace any compromised hardware if need be. We have decided to store the team's code in a GitHub repository which will allow our code to be protected in an off cite facility. The loss of developer computer systems is by far the biggest risk here, as our budget would not be able to cover the replacement of such computer systems.
RSK9	Especially in the current coronavirus climate, our developers may be unwell for a period of time that has a negative impact on the development of the project.	Within our schedule we have included time for slippage that should allow for any time needed by the team to be taken off due to illness. Should a developer need to self isolate and should they not be experiencing symptoms, they could continue to work on the product from a remote location using the GitHub repository.
RSK10	An Inadequate testing strategy could allow unidentified bugs to be released in the product when handing it over to the client. This could lead to a disappointed client.	To mitigate against this risk we have devised a testing strategy that ensures thorough testing is carried out throughout the development of our product. We are confident that the proposed testing strategy will allow the team to identify and correct errors in the system before the product is released to the client.
RSK11	Poorly developed code could mean that despite substantial testing, many bugs could still be included in the product at the conclusion of the project lifecycle, leaving our product to be ineffective for the uses that the client requires.	To mitigate this risk, we will be following our testing strategy outlined in this document to allow for integration testing. This means that testing will take place every time a new feature is added to the system, allowing us to detect bugs quickly as features are implemented.
RSK12	The wearable device we create may cause patients to feel uncomfortable limiting their use of the device.	We are slightly out of control with this risk as it mainly depends on how the patient reacts to the wearable. Having said this, we aim to make the wearable as discrete and as watch like as possible in an attempt to avoid the patient feeling uncomfortable when wearing it.

Continued on next page

Code	Risk	Mitigation
RSK13	Our device could startle or scare the patient putting them in danger.	We have taken on board the advice of the client for this risk and will be ensuring the watch does not use vibrations or LEDs unless the patient is deaf. We will ensure that generic alarms are not used also in an attempt to avoid startling the patient.
RSK14	One of our developers could leave the institution, lowring the number of developers we have available to work on the project.	Should this occur, we would definitely need to use the slippage time we have allowed for in our schedule. We would not be able to bring another developer into the team and would therefore need to share the workload of the developer that is leaving to the other developers on the team.

6.2 Risk Likelihood and Impact Matrix

Table 6.2: A matrix detailing the likelihood of a risk occurring along with the relative impact caused by that risk occurring.

Likelihood \ Impact	Low	Medium	High
Low		RSK2, RSK4, RSK7	RSK8, RSK13, RSK14
Medium	RSK12	RSK3	RSK1, RSK5, RSK10, RSK11
High		RSK9	RSK6