CSP400 Software Delivery Project

Narrative and Reflective Account

Benjamin Sampson
914545@swansea.ac.uk
May 8th, 2020

Contents

1 Narrative and Reflective Account

1.1 Introduction

In this document, I will be providing a narrative and reflective account on the entire development process for the Haptic Feedback application. I will be detailing any problems that have arisen including if they fell within my risk strategy and alongside mitigation steps that I used. I will also be discussing the general requirements, initial designs and the methodology that I used whilst detailing the tools used for development. Any challenges that I have faced during development will be discussed alongside a reflection and evaluation of my mistakes during development and with the overall application.

1.2 Development Process

1.2.1 Requirements

Before development, I organised meetings with my supervisor and the PHD student that this application will be for, to not only view the hardware that will use the application but also to gather requirements of the application. Over the course of a several discussions, we were able to come to the conclusion of specific, important functional and non-functional requirements that would allow the application to work with the hardware. The application would initially make use of 5 specific pages, but as further meetings progressed, this became 6.

Due to there already being example code that works for the hardware device, I had a good idea as to what my application would do to allow the creation of this code from a user's perspective. Each requirement that was discussed had a specification attached, which detailed how the requirement would be met and what conditions were needed to achieve this. Over the course of the project, these requirements were achieved along with other requirements being discussed and implemented at request of the supervisor, to help aid the user experience.

1.2.2 Design

Designing the application was something that had to be considered in different ways, due to the fact that my application would be for user use and will also be passed on for further development. Over several 'showcase' sessions with my supervisor, we changed the layout and design of the application in several ways to make it look more appealing to others. I tried to keep the design similar across all pages of the application where possible, and over several discussions we came to a finalised design.

Not only did I have to discuss UI design requirements, but also code design requirements. This was because I would be eventually passing my application and source code off, for possible further development. In order to achieve this, I had to ensure that my code was well documented, tidy and easy to understand.

1.2.3 Methodology

My chosen methodology for development of my application was using Agile. This was because it offered sprint cycles which allowed me to separate up the work in sizeable chunks, ensuring that development ran smoothly as well as there being time to test and obtain feedback on that specific area of development. Thanks to this, it ensured that I was working to the needs of not only the user, but also my supervisor, and could develop something that met all requirements and also had the ability to expand further. During the initial planning of these sprints, I was able to add in areas that were designated as 'padding' to account for any development issues, any risks that occurred and also time to add new features to the application.

Due to the potential size and scale that this application could become, along with the mandated time frame that I had to work with, I felt that this methodology was the best one to pick. The only issue that has occurred with using this was the slight difficulty in obtaining direct feedback from my supervisor during a pandemic This was mitigated by on-line meeting sessions where I could show the current stage of the application, and gather feedback. This was also accounted for in my risk mitigation strategy so I was able to for this during development.

1.2.4 Tools and Techniques

During the initial meetings with my supervisor and PHD student, it was agreed that I would develop the application using Android Studio and Java. This was because I was familiar with these programmes and also something that they knew as well. As this application would be passed on for further development, developing it in a way that would ensure a smooth transition between developers was ideal.

Thanks to Android Studio being focused towards development of Android and mobile applications, I was able to use pre-defined layouts to save time, along with creating application pages using their builder tool. This allowed me to create simple layouts for each application page, and link them with functionality code easily at the click of a few buttons.

A useful tool that I decided to use was an Android Studio plugin that helped me generate all class diagrams for my project. This automatically generated code and images on class associations, and included everything I would need. This was then used within my Design Document and saved lots of time. The tool that I used was "Graphviz"[1] which provided graphical representations of UML code that was generated automatically by "Sketch-It!"[2]. This generated UML code that worked in unison with "PlantUML"[3] of classes inside a file structure. As I initially split my code up into several folders containing all classes specifically for use in areas of my application, this gave me visual class diagram representations of these folders, the classes, methods and relations to each other. I did however, have to manually make a rough UML detailing the overall structure of my application, and how each main functionality class came together and build my application.

1.2.5 Timeplan Discussion

I developed a Gantt Chart to follow, which detailed the development cycle of my project, over the course of 4 sprints. These sprints were split up into different aspects, starting from the initial UI development, to ending at the main functionality of the application. Throughout the course of development however, I didn't follow the Gantt Chart designed, but instead decided to skip around different areas of my application. This was because if I developed the 'Frame Functionality' I could also develop the 'Saved Frame Functionality' after, which would allow for testing to be conducted. Also due to how the 'Saved Frame Functionality' and 'Saved Sequence Functionality' worked, it would be easy for me to develop one area after the other due to their similarity.

Due to not fully following the initial Gantt Chart designed, I fell behind the schedule I set myself, however I was able to put more work and effort into my application to bring myself back on track. Thanks to this, I was able to finish the rest of my application early, which gave me the ability to add and include new features and functionality, as well as rework the initial UI design at the request of my supervisor. Not only did this complete my Requirements and Specifications, but also allowed me to develop the application more a to the needs of the supervisor so they can conduct detailed user studies.

[1] Graphviz: https://graphviz.gitlab.io/download/
[2] Sketch-It!: https://plugins.jetbrains.com/plugin/10387-sketch-it-
[3] PlantUML: https://plugins.jetbrains.com/plugin/7017-plantuml-integration

1.3 Testing

When it came to testing my application, I employed the use of Gorilla Testing. This ensured that each new UI addition or change, would be tested against its required functionality on every new build of the application. As each UI element worked in one way or another with different areas of the application, I had to ensure that each change made to the application didn't break any UI functionality. Acceptance tests were the main focus of documentation and have been generated based on a pass or fail basis to specifications and associated requirements.

Following from Gorilla Testing, I conducted Unit Testing on each new method or functionality added. This constant testing of individual methods ensured that there were no bugs before I moved onto the next stage of development on that specific area of the application. I conducted these tests on a page by page basis, ensuring that everything on a page worked individually before moving onto the next area.

After Unit Testing on each area of the application was conducted and completed without errors, I moved onto Integration Testing. These tests ensured that each individual UI element that contained functionality worked with everything else on that specific page. This ensured that there was order to the applications pages, and each page worked as intended. If one UI element couldn't fully interact with another on the same page, I would attempt to fix the issue to ensure everything worked in unison.

System Testing was used to ensure that each page of the application worked together. This ensured that the initial Functional and Non-Functional requirements that were agreed upon, could be met as every area and aspect of the application worked with each other where applicable. Due to how my application works, Exporting Frames/Sequences, using both Frame and Sequence Creation areas of the application. It was important that each area worked together - System Testing helped to achieve this.

Finally, Acceptance Testing was conducted on every specification that was agreed upon, at the start of development. This ensured that every requirement had a working specification attached to it and that the overall quality and performance of the application was up to standard of both my supervisor and PHD student. It allowed me to come to the conclusion that I have met everything that was asked, and that the project was a success on my part. I also conducted Acceptance Testing on the new features that I included, at request of my supervisor, to help aid in usability.

The use of these testing strategies was important because of how different parts of the application interact with other areas of the application. It was also important due to the different user studies that will be eventually conducted using the application, ensuring that every area works smoothly and can be operated by someone who has never seen or used it before, was very important. These tests helped to achieve all of the initial requirements and specifications associated to the project.

1.4 Challenges

1.4.1 Saving Data

My application makes used of user generated information like '$F000' and '$N001'. In order for me to achieve proper verification within the application, to account for duplication or caps on specific choices they made, I chose to save the data to ArrayLists. This however, posed several challenges of its own. In order to conduct checks, I had to save specific commands to multiple ArrayLists then combine them at the end to create the finished sequence. I could then run checks whether an array has been filled or if it contained a specific character set. This posed an issue of multiple ArrayLists containing multiple characters that shouldn't be there, so a lot of formatting was required to remove this. Overall, I chose to use ArrayLists due to the simplicity of the saving technique, although not the best it is enough thanks to the temporary data the user is storing. As the user can load and export both Frames and Sequences, it allowed me to easily edit existing data alongside saved data.

1.4.2 Exporting Frames and Sequences

One of the biggest challenges that I encountered during development, came in the form of exporting my saved Frames and Sequence data. As already mentioned, I had to save many different forms of data inside different ArrayLists in order for the application to print the desired results. An issue that came with this however, was how Array's are printed with the inclusion of '[,]'. In order to print Frames and Sequences, without the inclusion of these characters, I tested two formatting methods. One method was manually selecting the information I required using their location index in the Array itself; this was done by using '.substring'. However, the main issue with this was it caused the exporting complexity to become massive, with slow build times that subsequently increased the length of my code massively.

The method that I found best, was using '.replace' to manually filter out all unnecessary characters from the Array string, so it can be formatted and printed in a readable format for the hardware to detect. This ensured that my code was shortened dramatically and also exported it in the way needed. As Sequences contain random Frames in any order, this had to be shown whilst exporting the sequence and was done using a similar method.

1.4.3 Granting Permissions

Due to my application saving files to a user's device as well as loading files, I had to figure out Androids permissions to give me the ability of read and write on the user's device. After following several tutorials, and creating methods in my program along with editing my applications manifest file, I was able to request both read and write storage permissions. This was a challenge as I initially couldn't test my exporting methods, due to not having granted permissions. After lots of trial and error this was achieved.

1.5 Technology Choices

1.5.1 Android Studio and Java

As already mentioned, Android Studio was used as the main development IDE. This was because of its great features and plug-ins that allowed me to easily develop UI and give functionality to my application. This worked hand in hand with the Java programming language, as it was something I was very familiar with, and it also meant that whatever I develop, can easily be passed on for further work by another developer proficient in the language. I chose using Java over Kotlin as the programming language, as I didn't trust myself to learn Kotlin in a short time, and ultimately develop a semi-complex application in a language I knew little about. Android studio offers the ability to convert Java files to Kotlin files - which means that code can easily be changed if future developers desired. However, as both me and the PHD student who would likely use the program know Java, we thought it would be best to use the language.

1.6 Mistakes and Issues

An issue that I initially had during development was how I was going to store data. I tested several different ways of potentially storing the user generated data, but ultimately decided to go with multiple ArrayLists. I chose this storage method as it was something I was very familiar with, and due to the fact that the data had to be exported, I felt it was the better option for temporary storage of data. When I added the ability to load in data from files currently saved on their devices, I was able to load the Frame or Sequence data directly into an ArrayList that could then be edited easily. The problem with this method however, is that the more complex my application became, the more ArrayLists I required for verification, exporting and editing.

Due to my application having the ability for users to use an 8x8 grid of check boxes to program individual pins, then have the ability to see which pins have been selected or not, I had to add the functionality for each pin individually. This caused an issue because the code of my program became quite lengthy, and very similar in some places. This was done by using a 'switch-case' statement, to introduce functionality for each check box individually. When it came to displaying Frames or Sequences on this 8x8 grid, it included multiple lines of code to detect and tick boxes according to previously used data. Once again, this added to lengthy run times and also hard to read code without proper commenting, but due to the nature of the application, was the only way to program the check boxes separately.

A finial and potentially significant issue that occurred during the finial development stages was the inability to test my application using its dedicated hardware that would work in unison with it [Due to COVID-19 and the pandemic it has caused, along with the closure of the university] The ability to access the hardware to test my application has become difficult, this means that although my application works and contains all requirements, there is no evidence that it works with the hardware it is designed for. Thanks to the uniqueness of this issue, there has been no mitigation steps or plan I could employ to navigate around this.

1.7 Evaluation

Overall, through the course of development I have encountered issues and made some mistakes. However, with the initial limited knowledge and overall time frame for the project, I think I have mitigated these in the best way I saw possible. There is plenty of room for not only future work, but also the ability to pass the application onto other developers.

When looking back at my initial risk analysis and mitigation plans, only a few risks ended up happening in a way that effected my development schedule. The main risk that I expected was "High workload", the assignment schedule I encountered throughout development meant that I had to manage my time, finish assignments early and dedicate time to other things outside of the applications development. I feel that I mitigated this risk well and it didn't cause as much of an impact as I previously thought.

A risk that I never initially foresaw was a pandemic, at time of writing previous documentations there were no issues. However, as the year progressed, COVID-19 spread and caused a lock down in the UK, which caused issues to the development and testing of my application, as well as minor issues to completion of other assignments. I managed to mitigate these issues by ensuring that all work was done before deadlines and splitting my day up into chunks where I could effectively work on different areas of my course.

When writing the Interim Report for the project, I was behind on quite a few of the requirements and functionality that is required in my application. However, work has been done to ensure that everything that was originally behind schedule, has been completed. This also ensures that I have caught up with my initial time plan that I developed and followed, with extra time to add in other features that were discussed to potentially enhance a user's experience with the application. One of these features comes in the addition of help icons for each area of the application, when the user clicks on one of these icons, they will be able to view information about that feature.

Thanks to this project, I was able to understand what it is like developing Android Applications for an external user, ensuring that I keep to good coding practices so that my code can be passed onto other developers without issue. I could also apply different risk mitigation strategies along with developing new ones, for areas of issue that I encountered during development. There is plenty of room for extra development within the application, and given enough time, I feel that I could develop it further. Due to the time-scale of this project and my initial uncertainty over developing applications, we chose to leave out some requirements that were not as important but would have been nice to include. However, the set of requirements we agreed on were discussed and completed within time, which allowed for the addition of new features.