Milestone 2 : Interim Report

Matthew Pike, 523355@swansea.ac.uk
Supervisor: Dr Max Wilson

Swansea University
Prifysgol Abertawe

Swansea University
Computer Science Department

Contents

Introduction

In this document we aim to report the progress that has been made in the development of the application outlined in Milestone 1. The document will analyse the development on the project and outline some of the techniques and approaches that were used to fulfil this work. Additionally we will document the remaining work left to do in order to complete the project.

As a part of Milestone 1 we produced a risk assessment document which documented the potential risks we may encounter during the construction of this project. Since a reasonable time has elapsed between the original risk analysis and now, we will revisit the original assessment and analyse whether or not it was a thorough and correct assessment. We will revise where necessary and present an updated table in this document.

Finally, we will look at the projects remaining time plan, accounting for any delays that have occurred so far. This updated timetable will serve as the plan for completing the remaining work of the project.

This document will serve as Milestone 2 for this project.

Project Summary

The aim of this project is to develop a software application that allows the Client to gain qualitative insights to interface designs with the use of commercially available brain scanning equipment. The focus for the this project is analysing web based documents, so standard web pages consumed through a web browser.

The first part of this project will aim to provide this browser, along with some additional functionality. The primary functionality of this modified web browser will be the ability to record data from many devices and user interactions. The primary source of data will be the Brain scanner, but additionally we will aim to capture data such as audio, screen-shots and general (user derived) browsing events. This modified browser could then be deployed during a user study in order to record the proceedings of the study.

The second part of the project focuses on the presentation of this collected data in an easily conceptualised manner. This visualisation aspect of the project will aim to allow Researchers to gain insight into the recordings through a visualisation.

Term Definition

As with any technical project, there exists numerous technical abbreviations that are used in place of long descriptors. We present a table below of the various abbreviations used in this document, which should serve as an aid to the reader.

Term	Definition
Visualiser	The part of the system responsible for displaying the data visualisation.
Document	The web site or web based document being investigated in the user study.
Experiment	The study that is being conducted. Typically will contain a User and a Conductor.
Data Source	A device/software/location that is recorded by the Web Browser.
Recorded Instance	Since a single experiment on a single user can contain numerous conditions and task, we must distinguish what one unique combination of these properties are called. We have chosen 'recorded Instance' to represent this. A Recorded Instance is an identifier for - Experiment -> User -> Condition -> Task.
Stack	A Stack is a part of the Visualiser UI that represents a single Recorded Instance. A Stack is therefore a UI component.
WPF	Windows Presentation Foundation - The visual framework used to develop the user interface(s) in this project.
ORM	Object relational mapping - a way of linking entries in a database to a usable format in application code.
SQLCE	Sql Compact Edition, the embedded database product we use to store application data.

Persona's

In addition to technical abbreviations, the project features unique individuals that partake or feature somehow in the system. Below is a table documenting these individuals and their role in the system. Again, this table is meant as an aid to the reader.

Individual	Role
Conductor	The person responsible for performing the user study. This is typically a researcher who is aiming to prove a particular hypothesis.
Researcher	The person who is responsible for gaining insight from the user study. The Researcher and Conductor can typically be the same person, however this is not always the case, especially in a large research group. The Researcher is therefore a member of the research team who is wishing to use the data collected from the study.
User	The person who is sitting the user study. Their responsibility is to perform tasks provided to them by the conductor.
Client	The person who will be receiving the finished software project. In this case it is Pingar.

1 Progress

1.1 Overview

In this section we will be discussing the progress that we have made with the project. The progress will be matched against the system specification and requirements, to make sure that the project is on target and not going off specification.

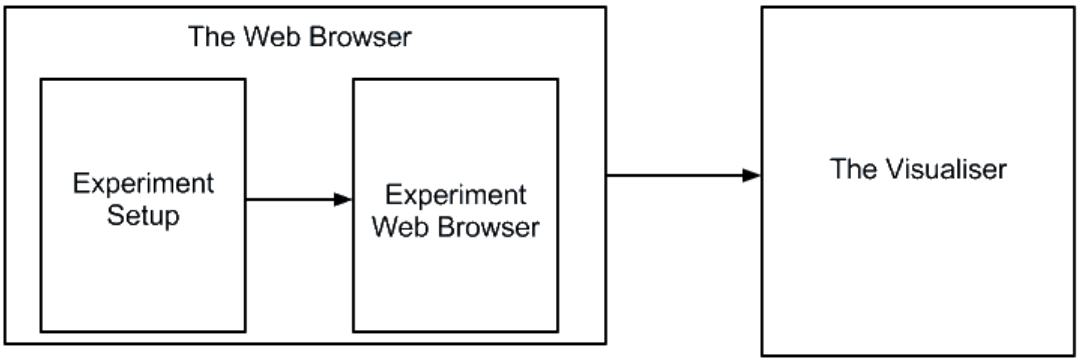

Figure 1.1: Component Interaction Diagram

Figure 1.1 provides a visual representation of the way each component interacts with the other. It is included here to give the reader an idea of each classes responsibilities in the overall project.

1.2 Part 1 : The Web Browser

Part 1 of the project, named "The Web Browser" comprises of two unique stages -

1. **Experiment Setup** - It is here that the conductor specifies the experiment, participant, condition and task that the experiment should operate upon.
2. **Web Browser** - The web browser serves two purposes. The first is the interaction front end for the participant to use to browse web pages. The second is the collection of the metrics generated from the browsing.

There has been significant progress in this part of the project. In fact the part is almost complete, with only certain documentary articles needing completion. We

include the table below to give a convenient breakdown on the status of the requirements for this part (in accordance with those identified in Milestone 1). For each requirement we include the related specifications, a status indicating progress and finally a brief comment on the requirement to give context and brief documentary notes on the implementation.

Requirement Id	Specification Id	Status	Comment
Functional Requirements			
WBFREQ1	WBSPEC1, WBSPEC2, WBSPEC3, WBSPEC4, WBSPEC5	Completed ✓	The conductor is able to specify the necessary experiment details via the user interface.
WBFREQ2	WBSPEC6, WBSPEC7, WBSPEC8, WBSPEC9, WBSPEC10	Completed ✓	The user interface maintains all of the previously entered data via an embedded database. The user interface then queries this database and displays the information on the user interface.
WBFREQ3	WBSPEC11, WBSPEC12, WBSPEC13 , WBSPEC14, WBSPEC15, WBSPEC15, WBSPEC16, WBSPEC17, WBSPEC17, WBSPEC19	Completed ✓	The user interface contains an experiment collectors setup page that allows the conductor to specify the relevant collection sources for the experiment. This data is also stored in the embedded database.
WBFREQ4	WBSPEC20, WBSPEC21, WBSPEC22, WBSPEC23	Completed ✓	The application allows full data management via an intuitive user interface.
WBFREQ5	WBSPEC24, WBSPEC25	Completed ✓	The application presents the conductor with additional dialogues in order to set-up and calibrate the collection sources.

WBFREQ6	WBSPEC26, WBSPEC27, WBSPEC28, WBSPEC29	Completed ✓	The web browser resembles a very basic, but feature complete web browser.
WBFREQ7	WBSPEC30	Completed ✓	The web browser uses IE9 as its rendering engine, and is therefore standards compliant.
WBFREQ8	WBSPEC31	Completed ✓	The web browser stores all data to file.
WBFREQ9	WBSPEC32	Completed ✓	The web browser contains a status strip on the bottom of the web browser alerting the conductor of the collection sources status.
WBFREQ10	WBSPEC33	Completed ✓	The web browser contains a finish button that safely closes the collection sources down and closes the output streams.
Non-Functional Requirements			
WBNFREQ1	WBSPEC34, WBSPEC35	Completed ✓	The web browser has been configured to disable scripting warnings and other visual clutter that could potentially interfere with an experiment.
WBNFREQ2	WBSPEC35	Completed ✓	The web browser is intuitive to use according to a group of beta testers.
WBNFREQ3	WBSPEC36, WBSPEC38	Completed ✓	The web browser successfully collects from numerous data sources concurrently without interference.
WBNFREQ4	WBSPEC37, WBSPEC39	Incomplete ✗ ~60%	The web browser is fairly well commented in code, but the process of commenting is not yet complete.
WBNFREQ5	WBSPEC40	Completed ✓	The web browser operates within these specified system requirements.
WBNFREQ6	WBSPEC41, WBSPEC42	Completed ✓	The web browser contains abstract classes and interfaces that allow for additional functionality to be added with relative ease.

1.2.1 Experiment Setup Implementation

In this section we will look at the experiment setup component of the application and discuss how it was developed.

1.2.1.1 Home Page

Figure 1.2: The initial starting page when running the application setup application

When starting the application the user is presented with the "Home" page which summarises the applications purpose and operation. The screen (shown in Figure 1.2) does not perform anything technically interesting as it is purely an information screen. From this screen, the user then moves to the next "Page" by clicking the desired page name.

1.2.1.2 Experiment Page

Clicking the "experiment" title loads the experiment page, shown in Figure 1.3 & Figure 1.4.

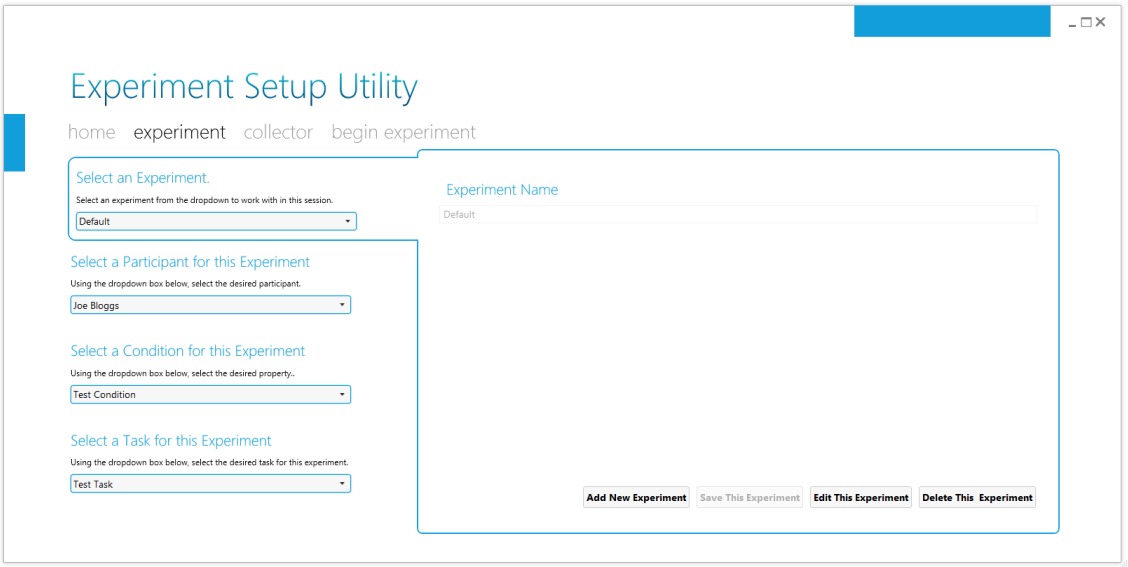

Figure 1.3: The experiment page of the setup application.

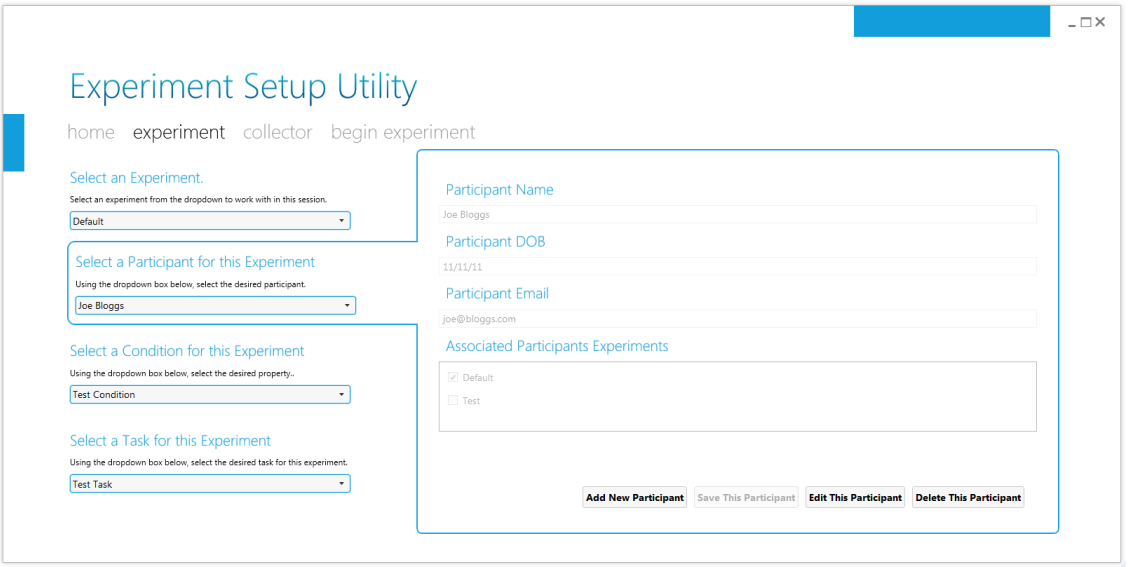

Figure 1.4: The experiment page of the setup application showing the participant selection and CRUD interface.

We see that the experiment page is in fact a vertical tab control with 4 tab items, each providing functionality to view, edit and delete - Experiments, Participants, Conditions and Tasks. This can be considered as fulfilling the CRUD aspect of the project since it allows for this operation on each one of these entities. By selecting a tab, the view is updated to display a form showing the related properties of the item selected in the tab. For example, when clicking the “Participants” tab, the

application will display the information of the currently selected participant (from the drop down box).

Each entity (Experiments, Participants, Conditions and Tasks) within the experiment page provide a form based interface for fulfilling the CRUD requirements upon each of them. The forms are fully validated when a change is made, and the user is alerted via a pop-up message if an entered value does not pass the validation. Additional prompts are included for destructive actions such as deleting an entity, confirming that this is the users intention.

To store the experiment data created in this experiment setup utility the application uses the Microsoft Entity framework to provide the Object-relational mapping (ORM) functionality. This allows us as developers to completely ignore the implementation details regarding the database setup and the necessary connections between the database and the application. The only thing we need to consider are the classes which represent the objects which we wish to store in the database. These classes were written in standard C# code. Using these classes the Entity framework is able to infer the necessary tables and relations. Figure 1.5 shows the relation between the tables within the database that our application is using. It is important to remember that this entire structure was automatically inferred by the Entity Framework.

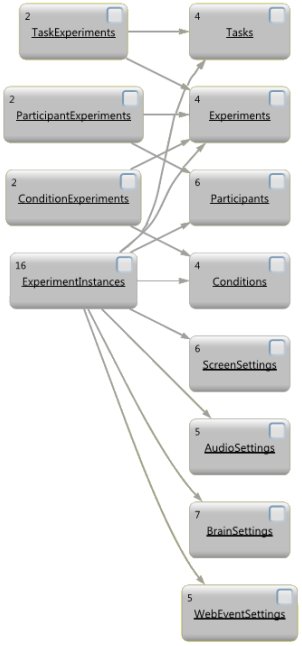

Figure 1.5: The database tables used to store the experiment information.

To interface with the database (which is created automatically by the Entity Framework) from our code we simply interact with a "mock" object, which is basically the data from the database in a List format. So to add an item to the database,

we simply add it to the list, and then save the list. A similar approach is taken for Read, Update and Delete. This approach greatly reduces the overhead of using a database in an application, but still provides us with the benefits of a database, allowing us to retrieve for example a Participant with a particular surname. Developing with the Entity Framework has been reasonably straight forward, we did have some issues initially using our desired Sql Database product - Microsoft Sql Compact Edition, but this was resolved with the inclusion of a recently updated driver (which is automatically included with the Experiment Setup application). As a reference for the Entity Framework we used the book "Programming Entity Framework" by J. Lerman [1]which provided useful practical examples to explain potentially complex concepts using the Entity Framework.

1.2.1.3 Collectors Page

By selecting the "Collector" title from the navigation menu the user is then presented with the display shown in Figure 1.6. The options presented allow the conductor of the experiment to specify exactly which data streams he or she wishes to collect during the experiment. As with the entered data above, the settings are stored in the embedded SQLCE database file. The interface additionally contains configuration dialogues which allow the conductor to calibrate the inputs sources appropriately. The implementation here follows that of the screens that have appeared before.

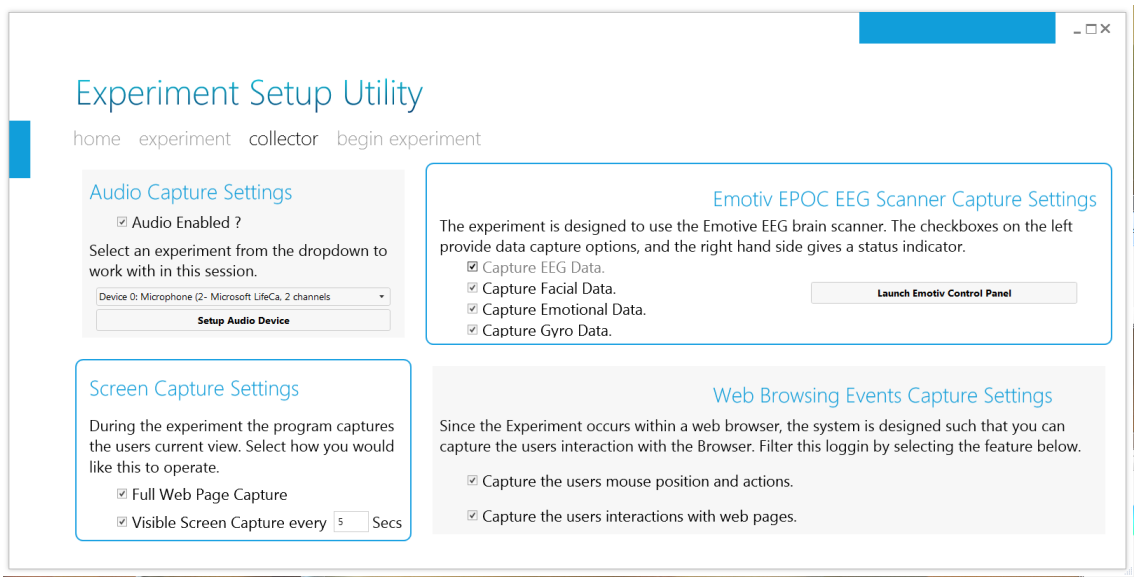

Figure 1.6: The Collector page used to setup the data sources to be recorded during the experiment.

1.2.1.4 Begin Experiment

The final stage in conducting an experiment is to finalise the settings and supply a starting URL, and then running the experiment itself. We see from Figure 1.7 that the page:

1. Summarises the chosen experiment properties via a scrollable text box
2. Allows the conductor to supply a starting URL.
3. Allows the conductor to either begin the experiment or exit the setup application.

Clicking the "Begin" button runs the Web Browser (documented below), which is provided with the experiment instances unique identifier. Using this ID the web browser is able to query the application database and retrieve the experiment settings using the ID. Again this is facilitated via the Entity framework, which also promotes code reuse since the same methods are called in the web browser as the ones used here.

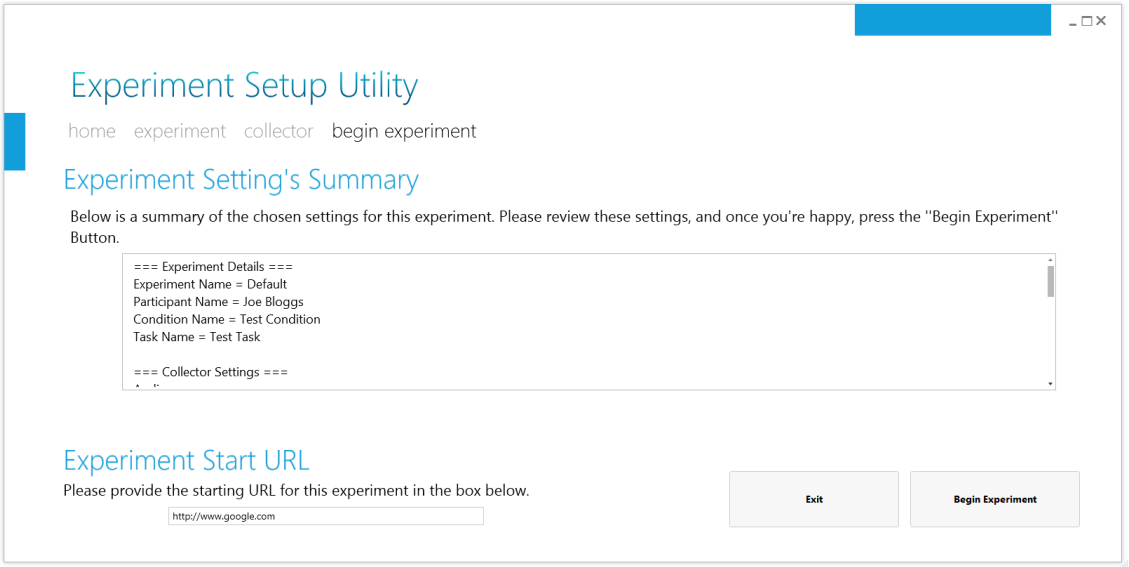

Figure 1.7: The final step in the setup utility. From here the conductor can exit or begin the experiment.

1.2.2 Web Browser Implementation

Having configured the desired experiment properties, the conductor is now able to begin the experiment. Once this operation begins, the web browser (shown in Figure 1.8) executes.

To the participant of the study, the web browser appears to be one that they are familiar to using on a day to day basis. This was our intention, and the primary

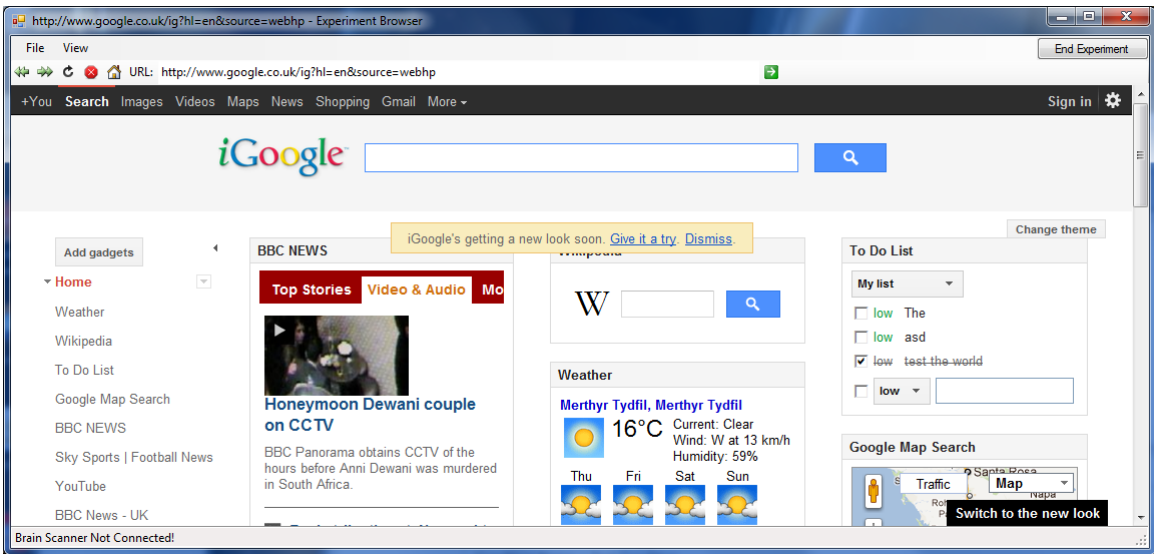

Figure 1.8: The web browsing interface the Participant will use during an experiment.

reason why the web browser appears to be very basic. To avoid "contaminating" an experiments findings, we attempted to make the web browser as minimal as possible, and yet familiar to a non-technical user. We feel that this is achieved with the implemented web browser. Of course there are 2 abnormal features that appear on the interface, that the participant will not be familiar with:

1. **Status indicator (Bottom Left)** - Indicating the status of the collection devices. This has minimal space on the display, since it is only really of interest to the conductor of the experiment, and not the participant.
2. **End Experiment Button (Top Right)** - Whilst fairly prominent, we feel the button will largely be ignored by participants since the purpose of the button is clear - it concludes the experiment. As such it is only of interest to the experiment conductor.

What is different about this web browser however is the process that occurs in the background, as the participant is using the Web Browser. Based on whether or not a particular functionality has been enabled or disabled in the experiment setup, the web browser will be collecting the following data:

Brain Data The web browser is responsible for establishing and maintaining a connection to the Emotiv Brain Scanner device. This is achieve through the standard API provided by Emotiv and documented in [2]. Having established this connection, the web browser then logs all the data to an XML file. To separate concerns, each unique data stream from the brain scanner is recorded in its own XML file, these streams are : Emotional data, EEG data and Gyroscopic Data. Each one of these files are then stored within a "Brain" subdirectory.

Audio The web browser also captures audio from the configured input device. This is again done without any additional setup or interference upon the user. Once the experiment is concluded the web browser saves the recorded file in ".wav" format and stores it within the "Audio" subdirectory.

Web Events As the participant browses the web, they generate interesting "trails", for example clicking a button on the page or highlighting text. The web browser captures all of these actions and stores them as events, within a XML file. This functionality is achieved by injecting a custom JavaScript library into each displayed page as they are loaded. The JavaScript library then registers its own custom callback methods onto every component on the page, and then listens for a user interaction. As an example, imagine the Google.com front-page, which contains an input box (the search query) and a submit button (Search Button). Once this page is loaded, our JavaScript library is dynamically injected into the page, and "attaches" (or registers) its custom listening methods to the text-box and submit button. When the participant enters a query, the library captures the users keystrokes and passes them to the web browser via a Common Object Model (COM) interface. Having done this, the user then clicks submit, which is again registered by our JavaScript library and passed to the web browser. On the web browser side, when it receives a web event from the JavaScript library, it simply logs the event in the XML file and awaits more events. These web events are then stored within the "WebEvents" subdirectory.

Screen-shots Finally the web browser captures screen-shots of the web browsers state at two distinct intervals.

1. On a page load - When a web-page completes loading, the web browser automatically captures the entire web page.This gives the conductor a full page reference of what the user would see.
2. On a specified interval - Set out in the experiment setup, the web browser also captures the visible area of the display in order to reference exactly what it was the participant was viewing at that period in time. The distinction between the two is made in order to reduce the size of the images captured at a set interval.

These images are then stored in the "Images" subdirectory.

When the web browser is operational, the collected data is stored in a temporary directory on the machines hard disk. Once the experiment is ended, this temporary data is converted automatically into a zip file. This approach was chosen since much of the data is sparse text files, so the compression is particularly effective on the large XML files that are generated. The Zip files are named according to the experiment instance ID that was used to initiate the web browser. This ID ties the recordings to the experiment instance in the database. Should this operation complete successfully and the .zip file is created, then (and only then) the temporary directory is deleted.

Once the "End Experiment" button is clicked, then the web browser gracefully exits, closing all output streams. We note that we have paid particular attention to IO efficiency, especially as we have a number of output streams from the application, writing often simultaneously, large amounts of data. To reduce this overhead, we researched the most efficient XML output libraries available, and discovered the built in "XmlWriter" included with the .Net standard library to be the most efficient. This efficiency does however come at the cost of usability, and many helpful library methods are missing from the library. To help with using this XmlWriter interface, we used "C# XML: Essential XML Skills" [3] which was a invaluable resource.

1.3 Part 2 - The Visualiser

In this section we look at the progression on part 2 of the project - The visualiser. This section has received significantly less attention than the Web Browser section above, and as such will be significantly shorter. We do however begin (as above) with the progress table below, outlining the current state of this section.

Requirement Id	Specification Id	Status	Comment
Functional Requirements			
VSFREQ1	VSSPEC1	Incomplete ✗ ~30%	The ability to select specific experiments to visualise is not yet fully implemented. Currently the only implementation detail that is concrete is the ability to load experiments and their associated detail from the embedded database.
VSFREQ2	VSSPEC2, VSSPEC3	Incomplete ✗ ~10%	The ability to stack visualisations in order to compare one another is not yet implemented. We have however researched some existing frameworks that will facilitate with the implementation of this functionality.
VSFREQ3	VSSPEC4	Incomplete ✗ ~70%	Our prototype is using the time-line control provided by Google [4]. The captured emotion data from the brain scanner is then displayed using the time-line control. To complete this requirement, the control needs integration to the other visualisation components.

VSFREQ4	VSSPEC5, VSSPEC6, VSSPEC7, VSSPEC8	Incomplete ✗ ~70%	Again the Google time-line control is being used to display the web events on top of the emotional data plot. To complete this requirement, the control needs integration to the other visualisation components.
VSFREQ5	VSSPEC9	Incomplete ✗	The ability to scroll through visual data is not yet implemented, since the underlying visualisation is not yet complete.
VSFREQ6	VSSPEC10	Incomplete ✗	The ability to detect visual correlations in the data is not yet possible, since the underlying visualisation is not yet complete.
VSFREQ7	VSSPEC11	Incomplete ✗	The ability to correlate different time regions in the visualisation is not yet possible, since the underlying visualisation is not yet complete.
VSFREQ8	VSSPEC12	Incomplete ✗ ~10%	The ability to add annotations to the events data is not yet implemented. We have however planned how this functionality will be implemented.
VSFREQ9	VSSPEC13	Incomplete ✗ ~10%	The ability to extract desirable signal values is not yet implemented. We have however planned how this functionality will be implemented.
Non-Functional Requirements			
VSNFREQ1	VSSPEC14	Incomplete ✗	This requirement is dependant on the entire project being finished.
VSNFREQ2	VSSPEC15	Incomplete ✗	This requirement is dependant on the entire project being finished.
VSNFREQ3	VSSPEC16, VSSPEC18	Incomplete ✗ 10%	This requirement is dependant on the entire project being finished. Some of the existing code base is very well documented.
VSNFREQ4	VSSPEC19	Completed ✓	The visualiser operates within these specified system requirements.

VSNFREQ5	VSSPEC17, VSSPEC20	Incomplete ✗	This requirement is dependant on the entire project being finished.

We see that this section is significantly less progressed than the previous section. This lack of progress is however within 1-2 weeks of our original time-plan and should not therefore be considered a major concern. Whilst a significant amount of work remains, we do feel we are in a position to complete the project on time. For further details on the project time-plan, see chapter 5.

There has been some progress however, whilst in prototype form, the progress is demonstrated in Figure 1.9.

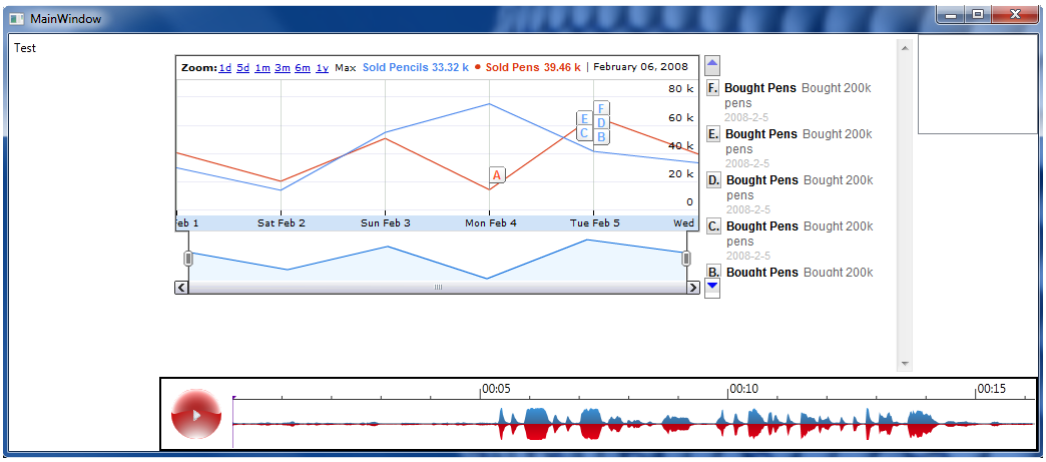

Figure 1.9: The current progress of the Visualiser.

We see from the prototype that two aspects of this part have begun:

1. **Audio Display** - We have implemented the wave form view on the display. For a given experiment instance ID, the audio display shows the Wave form for that experiment. To achieve this we use a 3rd party audio library - Bass.net (http://www.un4seen.com/) which provides us with the key audio amplitude data. This data is then plotted on a WPF chart to provide the view seen in Figure 1.9. The audio control is not yet operation e.g. the play button.
2. **Time-line control** - We have begun experimenting with the time-line control, as seen in Figure 1.9. We are using the Google time-line chart tool to display the collected brain wave data. Whilst the tool provides fantastic results, integrating it within the application has been non-trivial. In order to display the chart we have had to embed a web browser within the display and inject the time-line control dynamically. Far from an ideal approach, but we are now in a position where the time-line is working and we have added additional methods that allow us to add points and events to the chart.

3. **Data Processing -** Although not visible, the prototype contains the necessary methods to extract the saved experiment from its native zip format into an "in-memory" state. This functionality is essential when going forward and developing the remaining parts of the visualiser.

The window displayed in Figure 1.9 is representative of a single experiment visualisation. In our final application, a researcher will be able to have many of these windows open, and "stacked" on top of each other in order to easily compare experiment data. We see that the window layout is complete, but lacking functionality. On the left hand side of the window, space is reserved to include a the heat-map and mouse trail data. On the top right of the window is space for the researcher to select which data is to be displayed in the heat-map.

To complete the construction of this part, a "master" window needs to be constructed which will provide the researcher with the functionality to select the experiments they wish to view. The master window will also contain the experiment windows (seen above).

The primary challenge in this section will be creating a user interface that is flexible and intuitive to use, but also reasonably simple to specify and construct. We will likely be using the WPF framework as we have done in Part 1. As a point of reference, we have been using the book "Windows Presentation Foundation" by A. Freeman [5].

1.4 Summary

Having completed part 1 of the project, the focus going forward is almost exclusively on part 2. We are confident that our chosen methodology of Prototyping has worked well on the development of the first part of the project. We are continuing to use this approach for the second part of the project (as demonstrated in Figure 1.9). Whilst a significant amount of work remains to be completed on the project, we feel we are in a position to complete the project on time and to the necessary specification.

We conclude this section with a brief note on our development environment setup. To develop the application we are using C# on the .Net platform (Version 4). We use the LINQ extension language to perform dynamic querying upon supported data types (of which the Entity Framework is one). We use Visual Studio Ultimate 2010 as our IDE. To verify database data we use the Sql Compact Toolbox (http://sqlcetoolbox.codeplex.com/). Finally we use ReSharper for additional functionality in Visual Studio.

2 Remaining Work

2.1 Overview

In this section we will document and discuss the work remaining that requires completion in order to conclude the project in its entirety. The remaining work is organised in a tabular format for convenience.

2.2 Part 1 - The Web Browser

The table below outlines the remaining work required in order to complete Part 1 of this project. This should serve as a reference for the future efforts in the development of the project. Since the majority of Part 1 is complete, this section is fairly short with only a single remaining requirement to be completed.

Requirement ID	Specification ID	Task	Comment
Non Functional Requirements			
WBNFREQ4	WBSPEC37, WBSPEC39	The thorough documentation of the Web Browser.	As mentioned above, the majority of the Web Browser code-base is well documented, however we need to combine this documentation in with the documentation generated from the remaining section (part 2), in order to create a complete design document for Milestone 3.

2.3 Part 2 - The Visualiser

The table below outlines the remaining work required in order to complete Part 2 of this project.

Requirement Id	Specification Id	Task	Comment
Functional Requirements			
VSFREQ1	VSSPEC1	Add Experiment instances to be displayed.	This functionality is partially implemented from our development in part 1 of the project. The required database store and querying is working, and to complete this requirement we simply need to add an interface to access the stored data.
VSFREQ2	VSSPEC2, VSSPEC3	The ability to stack visualisations on top of one another.	The ability to stack frames has been researched and potential frameworks identified ([6]). Implementing this functionality is far from trivial but should be achievable thanks to the great resources and tutorials that exist.
VSFREQ3	VSSPEC4	The visualisation of brain data.	We have identified the tool we will use to visualise the data, and have begun prototyping with the time-line. The remaining work is to read in the recorded brain wave data and pass it into the time-line for visualisation.
VSFREQ4	VSSPEC5, VSSPEC6, VSSPEC7, VSSPEC8	The visualisation of web events.	Again this functionality will use the same time-line as above.
VSFREQ5	VSSPEC9	The ability to navigate through visualised data based on time.	The time-line we are using has this functionality built in. In order to fully utilise it we need to implement a listener that alerts the application when the user selects a particular region in the time-line, such that other components (e.g. Audio) can be updated.

VSFREQ6	VSSPEC10	The visualisation must be clear and comprehensi- ble.	The approach and design we currently have planned should allow for this requirement to be met, we cannot satisfy this requirement however until the visualiser is completed.
VSFREQ7	VSSPEC11	The visualisation must be linked to the other displayed data sources by time.	In order to fulfil this requirement the construction of a time class that is shared amongst the various components is required. This should be fairly trivial to implement using the Observer design pattern [7].
VSFREQ8	VSSPEC12	Allow researcher to add custom events.	A dialogue must be added to allow researchers to add custom events. This is fairly trivial and should be easy to implement.
VSFREQ9	VSSPEC13	Allow researcher to extract desirable boundary values.	A dialogue must be added to allow researchers to extract desired boundary values.
Non Functional Requirements			
VSNFREQ1	VSSPEC14	The visualiser must be intuitive.	This requirement can only be judged when the visualiser has been completed, and the application is evaluated by users.
VSNFREQ2	VSSPEC15	Visualisation must be informative.	This requirement can only be judged when the visualiser has been completed, and the application is evaluated by users.
VSNFREQ3	VSSPEC16, VSSPEC18	Must be thoroughly docu- mented.	This requirement can only be completed when the Visualiser has been completed.

VSNFREQ5	VSSPEC17, VSSPEC20	Handle vast amounts of data.	This requirement can only be judged when the visualiser has been completed, and the application is thoroughly tested.

3 Specification Analysis

3.1 Overview

In this section we will be analysing the specification we outlined in our earlier document. The aim of this is to identify any points in specification where alterations or clarifications need to be made and to identify how closely the specification has been followed. This section has subsections for each of the sections of the project been undertaken and each section will deal with the specific specifications related to their respective sections of the project.

3.2 The Web Browser

In this section we will aim to clarify some specifications relating to the web browser aspect of the project.

WBSPEC31 For this specification we stated that the captured data will be stored using a custom file format. We additionally stated that this format must be observable using standard compliant, existing tools. The clarification we wish to make here is that the custom file format is in fact a zip file which contains many sub-folders (related to the type of captured data e.g. "Audio"). Within each of these sub-folders are typically XML documents (where data is in text format) containing the captured data. These XML files are viewable using any standard web browser or text editor. For audio, we capture to ".wav" files which is royalty free and cross platform.

WBSPEC32 We stated that the web browser will contain a status panel updating the conductor of the collection sources availability. The clarification here is that the status only updates when a collector goes offline. Therefore the status of each individual collector is not shown, rather the status only displays notifications of collectors that have gone offline.

3.3 The Visualiser

In this section we will aim to clarify some specifications relating to the visualisation aspect of the project.

VSSPEC2 We state in this specification that each visualisation will present a recorded instance in its own individual stack. We wish to clarify the term "Stack" which has an ambiguous meaning in the original specification. In this context, we mean "Stack" to represent a single document within a multi document interface (MDI), with each document being representative of a recorded experiment session.

VSSPEC4 In this specification we stated that each "Stack" would contain a time-line display with amplitude data. We wish to add that as well as this amplitude brain scanner derived data, the graph will also contain a note of a web event occurring. This will allow the researcher to correlate certain web events to particular spikes in brain wave data.

VSSPEC11 This specification stated - "The visualiser will link all displays to the central control, which will b e t he time-line". This was somewhat ambiguous, and it simply attempted to convey the fact that the displayed information on the Audio display, interface display and time-line were relative to the time selected by the user in the time-line.

4 Risk Analysis

4.1 Overview

In this section we will be looking at the projects associated risks and what part they have played thus far into the development cycle. We will begin by analysing and discussing the risks we have encountered thus far in the development of the project.

Following this, we discuss the Risks that we have not yet encountered, and re-evaluate their applicability to the project. Additionally we will consider the possibility of the risks removal or modification to better represent the risks relation to the project.

This is then followed by an updated and complete risk analysis table (originally documented in our Methodology and Risk analysis document). This includes the updated scores, additional metrics and updated entries.

4.2 Encountered Risks

In this section we will record the risks that we have encountered up to this point in the development of the project. We discuss the effect the risk caused upon the development of the project and how we counteracted this effect. The risks are referred to by their unique identity (as used in the Methodology document in Milestone 1), but we also provide a quick summary of each, for the readers convenience.

Risk 5 - "High coursework load"

Since the development period for the project coincides with regular lecturing periods, the coursework load has been extremely high over the past few months. We originally predicted that the likelihood of this risk occurring being high (a score of 7) - we believe this score to be fairly accurate of the actual coursework load we experienced. We noted that our level of control to also be fairly high, with another score of 7, however we may have underestimated this factor. The sheer amount of coursework has put an intense strain on this project, however we feel that changing this rating now is somewhat unnecessary. The reason for this is that only one other

project remains to be completed. This cannot be underestimated however since it is a sizeable group project, and as such some risk can still be encountered from it. As such we have decided to keep this rating unchanged from that in the original document.

Risk 7 - "Company Demands"

As with any industrial based project, communication between the company and the developer is crucial. During the early stages of the development we had to manage the companies demands/expectations in order to ensure that the project was completed on time and represent a usable final product. Managing this balance between the features the client wanted and what was possible in the project time-frame was a non-trivial task, further complicated by the geographical separation between the client and developer.
We originally scored our ability to control this risk at 9. This high score was justified by the fact we were confident that through quality and continued conversation that we would be able to establish a fair compromise in functionality. However what was underestimated was the likelihood of this risk occurring, which was scored at 3. In hindsight it was almost inevitable that the company would want as much as possible out of the project and that we would have to manage these expectations. However, we feel that going forward, this risk has been alleviated and that the understanding between client and developer has been attained. As such we are removing this risk from the updated risks table.

Risk 8 - "Failure to judge time and resources"

Approaching the final phases of this development increases the risk posed by poor judgement, especially relating to time constraints. In our original analysis we scored this severity at 7, representative of the fact that poor judgement with regards to time is always critical. We feel that the original score was justified, however the time that has elapsed between the original scoring and now has made us re-evaluate and update this score to a 9. Our reasons for doing so simply represent the reduced time-frame within which we have to operate. Additionally, we originally scored the likelihood of this occurring at 4. We believe this score remains relevant today since we are actively working on the project and are aware of the necessary time investment required to complete this project.

4.2.1 Risks Encountered Outside of Risk Analysis

In this section we will look at some of the risks that have been encountered, that were not documented in our original risk analysis.

Risk - "Poor communication setup"

Since the client (based in New Zealand) and the developer (based in Wales) are geographically remote, we communicate via the P2P chat application Skype. This is not perfect however and it can regularly be difficult for one party to hear the other. The risk here is that an important bit of information is misheard or misinterpreted. The likelihood of this risk is fairly high, since it has been occurring during our meetings, however we combat the risk by summarising each meeting with a follow up email to ensure that we understood the topics of the meeting. This is therefore fairly low risk, but is worth documenting since it could be a catalyst for a variety of other risks.

4.3 Other Risks

Our original Risk analysis described a total of 12 potential Risks. We have however only encountered 3 of these. Below we look at the other risks that appeared in the original risk analysis and comment on their relevance to the project going forward.

Risk 1 & 2 - "Short / Long Term illness"

Should the developer of the project fall ill then it is clear that the project schedule will not be fulfilled and the project will overrun. This is especially prevalent in the case of a long term illness, which could see the project being abandoned all together. At this stage in the project however, we feel the effects of a long term illness will be inseparable from those of a short term illness (as a result of the now reduced operating time-frame), and as such we have removed it from our risk analysis. Short term illness remains however, and its effects are more prevalent now than at any other point in the projects development cycle. We originally scored the likelihood of this risk at 4, citing the developers current health and lifestyle being relatively good. This score remains (thankfully) realistic at this point in the project. The same is also true for our level of control. What is vastly different however is the impact of this risk upon the project at this point in the development cycle. We originally scored the severity of the risk at 7, but we have decided to increase this to a score of 8 to represent this additional potential effect.

Risk 3 - "Poor Time Management"

Similarly to Risk 8, described above, the failure to manage the remaining time could result in catastrophic consequences for the projects success. We feel our original scoring accurately reflected the risk at that period in time, we again feel however that we are required to update the severity of this risk to take into account the reduce operating time-frame. As such we are increasing the severity rating from a 7 to a 9.

Risk 4 - "Avalanche Effect of missed dead-lines"

This risk was originally discussed in relation to a number of big deadlines being missed and pushing the project beyond the point of completion (on time). This risk has not occurred (as discussed in Chapter chapter 5), and we feel that the project is sufficiently mature for this risk to be removed. Our reasoning is that the risk is fairly redundant now since its scope is covered by previously described risks e.g. Poor Time management.

Risk 6 - "Company Loses Interest"

The risk of the company losing interest in the project is a serious one, that can still affect the course of the project during these late stages. We feel that the likelihood of this risk occurring is very low since we have established a good dialogue between ourselves and the client. We still have to account however for this risk, as such we are updating two scores. Firstly we will reduce the likelihood of the risk from a 3 to a scoring of 1. Secondly we will increase the severity of the risk from a 8, to a 9.

Risk 9 & 10 - "Significant changes of client requirements , Poor understanding of requirements"

We feel that, at this stage in the projects development that no further changes to the clients requirements will be made, and that we understand the clients requirements well. We believe this as a result of the feedback we have received during meetings between ourselves and the client. We also feel that this well established communication and mutual understanding of each parties time constraints will mean that the client will not make changes to the project requirements. As such we are removing this risk from the updated risk analysis.

Risk 11 - "Lack Of technical Skill"

We have seen in chapter 1 that significant progress has been made on the development of the project so far. The chapter also describes the numerous new technologies that the we as developers have had to learn and adapt to in order to complete that section of work. It is also likely that the remaining work will require some technical skill or knowledge that we do not currently posses, so it is critical that we identify these skills and put measures in place to familiarise ourselves with them. We will therefore update the severity of this risk from a 4 to a 6 to represent the reduced time-frame for completing this project.

Risk 12 - "Device Failure"

Finally, we must re-consider the effect of the brain scanner suffering a hardware malfunction. Like many of the risks above the severity of a device failure is further compounded by the amount of time remaining to complete the project. We have updated the risks severity score to represent this. We feel the rest of our original risk analysis was and remains correct.

4.4 Updated Risk's Table

In this section, we provide the updated risk analysis table based on the discussion in the previous two sections. We use a similar table layout to that used in the Risk Analysis in Milestone 1, with an additional metric of Risk Priority Number (RPN). We summarise each metric below for the readers convenience.

- *Likelihood* - The likelihood of this this risk occurring
  - 0 - Not Likely
  - 10 - Very Likely
- *Severity* - The impact that this risk occurring could have on the overall project
  - 0 - Not a great impact overall.
  - 10 - A significant impact on the project, which would almost certainly derail the project.
- *Level of Control* - A measure of how much control we have over his risk
  - 0 - No Control whatsoever
  - 10 - We have complete control over a particular risk.

We additionally calculate the significance of each risk. This is in order provide a single metric for comparing each risk against. Significance is calculated by the following formula:

(Likelihood+ Severity) – Level of Control

This gives a single metric against which we can use to compare each risk relatively.

Risk Priority Number (RPN) is a measure used when assessing risk to help identify critical failure modes associated with your design or process. Typically used in Failure mode and effects analysis (FMEA), RPN plays an important role in evaluating system risks relatively. The RPN values range from 1 (absolute best) to 1000 (absolute worst). RPN is a standard risk measure that is used in numerous process industries worldwide. The metric is simple and easy to calculate, using the formula:

Severity * Occurrence * Detection

The scores are somewhat subjective, but have proved very helpful in producing a relative risk indicator for this project. We have decided to include the metric since it is widely used in industry and serves as a relative comparison between all risks.

Below is the table detailing the updated risk analysis for the remainder of this project.

Risk Id	Project Area	Title	Description	Likelihood	Severity	Level Of Control	Significance	RPN	Risk Stratergy	Risk Mitigation
RISK1	Personal	Short term illness	Should the develop er fall ill for a brief period, then he will have limited contact with the project during that period.	4	8	1	12	32	Careful Time Planning	We hope to overcome any short term illness by adding contingency periods to the plan.
RISK2	Personal	Poor Time Management	If the develop er fails to manage his time properly then it is likely that the project will overrun.	4	9	7	6	252	Regular progress meetings	By conducting regular meetings between the develop er and his supervisor and clients, the project should remain focused and be delivered on time.
RISK3	Personal	High Coursework load	The developer has one other significant project that will also require a good amount of his time, thus reducing time spent on this project.	7	6	7	6	294	Considered Planning	This risk can again be mitigated through careful and considered planning. By having a realistic time plan we can hopefully manage this risk.

RISK4	General	Company loses interest	Should the company lose its interest in the project, it would leave the develop er with very little motivation to continue.	1	9	9	1	81	Communication with the client.	Keeping the client informed with regular meetings will ensure that they maintain their interest.
RISK5	Judgement	Failure to judge time and resources	Should the developer fail to judge the projects requirements in terms of time and complexity.	9	7	7	9	441	Methodology	Since our chosen methodology requires regular iterations of the prototype, we are confident that we can mitigate this risk by complying with the methodology.
RISK6	Technical	Lack of technical skill	Since this project involves numerous programming interfaces and interaction with new technologies, it is possible that the developer will not be sufficiently experienced with these technologies.	5	6	8	3	280	Methodology Choice	We believe that our chosen methodology of prototyping will allow us to discover the technical requirements of the project early on in the development cycle. This will give the develop er sufficient notice of what his technical knowledge needs to b e.

RISK7	Hard-ware	Emotiv EEG Device Failure	Since the project is dependant on a hardware device, its failure could cause a serious setback in the projects progress	4	8	4	8	128	Taking care with the device.	The headset is designed for gaming, so is somewhat ruggedized but care must still b e taken. Additionally the department has an additional headset which can be considered a backup
RISK8	Hard-ware	Poor Commu-nication Setup	The hard/software set-up that the client and developers use to communicate with one another could be of poor quality.	7	4	5	5	140	Meeting sum-mary follow-up	The risk can be mitigated by conveying a summary of the meeting in text form to ensure all parties have understood one another.

5 Updated Time Plan

In this section we will compare the projects development to date against the original project time-plan, as documented in Milestone 1. The primary purpose of this section is to present an updated time plan which will chart the projects final few stages.

In terms of general progress the project is fairly on schedule. Each section has slipped slightly behind the original schedule by approximately 1-2 weeks. We do however feel that we are able to accommodate this delay in our updated time-plan. Before doing so, we must first pin point the reasons for this delay.

The primary cause of the delay has been the sheer lack of time that has been available to work on the project. The original Risk analysis predicted this as a potential outcome. The majority of our time has been invested in completing the significant amounts of coursework that have been set in parallel with this project. The good news going forward is that all coursework assignments have been completed and we are only tied to one other (albeit significant) group project.

Another cause for the delay was the chosen methodology for developing the project. We chose the prototyping methodology, which incorporates a number of small, disposable prototypes being developed before creating the final version. We feel that we iterated one too many times in this methodology and we should have realised that too much time was being invested in the prototyping stages of the development. We will need to bear this in mind moving forward.

The Gantt chart in Figure 5.1 shows the original time plan which was created for Milestone 1. It shows the amount of time that we estimated each section would take to complete. Figure 5.2 shows the same Gantt chart, but with the new estimated times for each section of the project. Finally, as these charts can be hard to compare directly, Table 5 shows a table which allows easy comparison between the old and new estimated dates of completion.

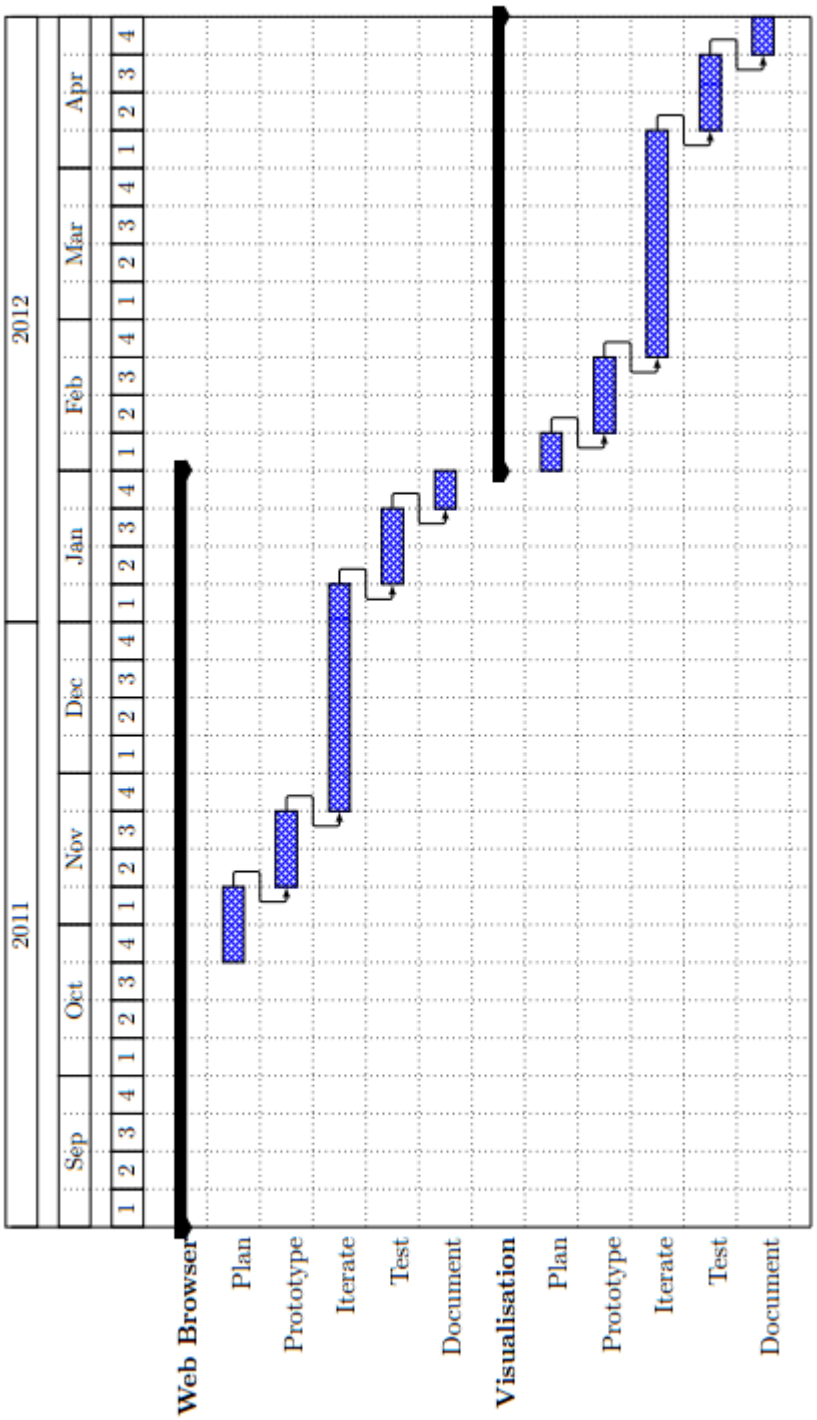

Figure 5.1: Gantt chart showing our original planned time investment.

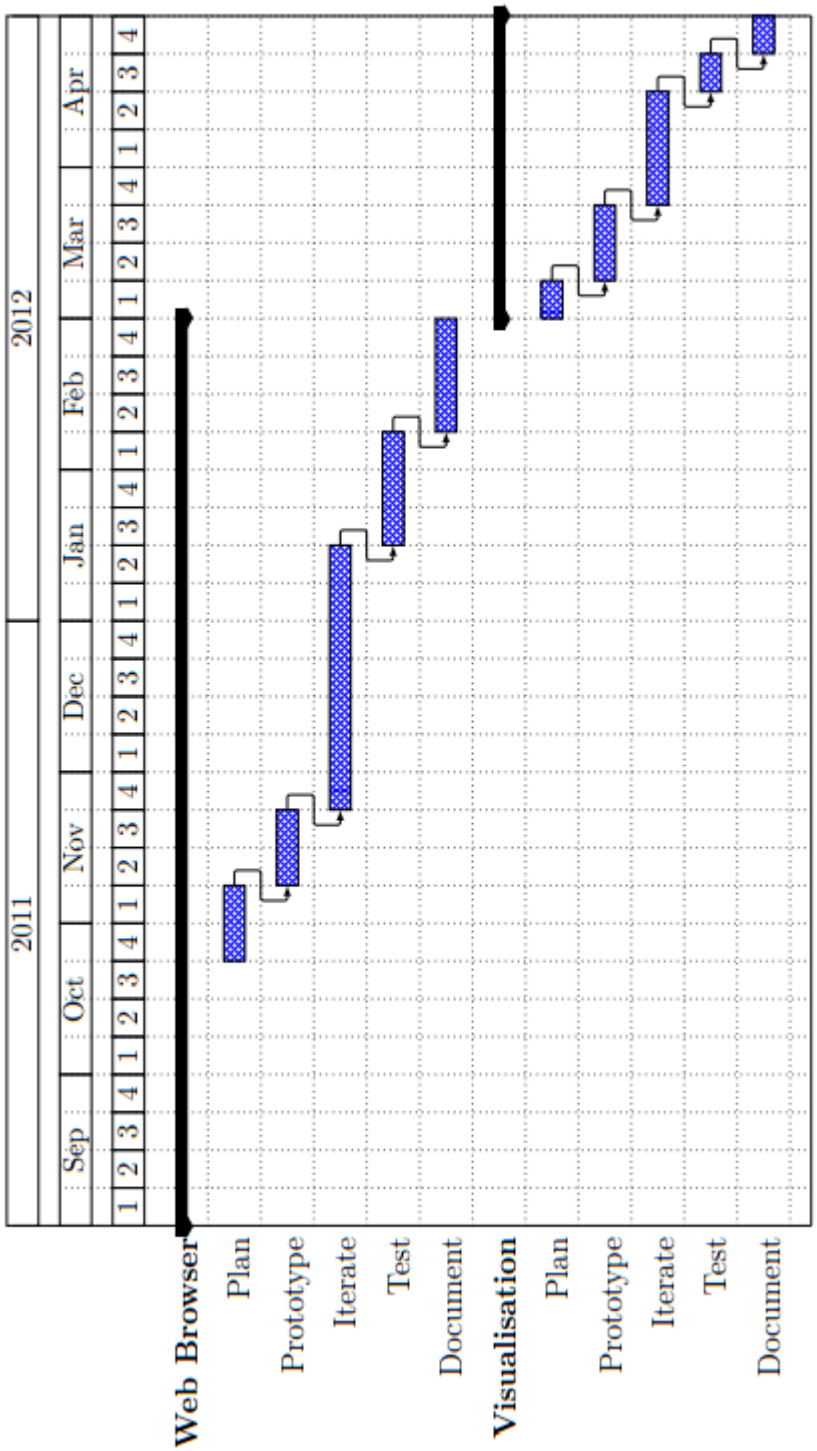

Figure 5.2: Gantt chart showing our updated planned time investment.

Activity	Start Date	Old End Date	New Start Date	New End Date
Web Browser				
Plan	27 Oct 2011	7 Nov 2011	27 Oct 2011	7 Nov 2011
Prototype	7 Nov 2011	21 Nov 2011	7 Nov 2011	21 Nov 2011
Iterate	21 Nov 2011	7 Jan 2012	21 Nov 2011	14 Jan 2012
Test	7 Jan 2012	21 Jan 2012	7 Jan 2012	2 Feb2012
Document	21 Jan 2012	28 Jan 2012	2 Feb 2012	28 Feb 2012
Visualiser				
Plan	7 Feb 2012	10 Feb 2012	28 Feb 2012	5 Apr 2012
Prototype	10 Feb 2012	21 Feb 2012	5 Apr 2012	12 Apr 2012
Iterate	21 Feb 2012	7 Apr 2012	12 Apr 2012	20 Apr 2012
Test	7 Apr 2012	21 Apr 2012	20 Apr 2012	22 Apr 2012
Document	21 Apr 2012	28 Apr 2012	22 Apr 2012	25 Apr 2012

Table 5.1: Comparison of the old and the newly updated completion dates.

We see that Part 1 (The Web Browser) of the project overran by approximately 2 weeks (ignoring the document aspect of the project). The reason for this overrun has been the massive amounts of coursework that we have encountered during the development of the Web Browser. It is now complete and the documentation aspect as been done in parallel with the second part of the project.

As a result of this overrun in part 1, the second part of the project (the Visualiser) needs to recoup this time somehow. To achieve this we have reduced the amount of time dedicated to "iterating" during the development cycle. This means that we won't be able to experiment with various approaches or techniques as much as we could have otherwise. This is not necessarily a negative however, since we described this "over-iterating" as the cause of the initial overrun.

We feel that the revised time-plan is both realistic and complete. We are confident that, despite this slight setback, the project will be completed on time and to specification.

Bibliography

[1] J. Lerman, *Programming Entity Framework: Building Data Centric Apps with the ADO. NET Entity Framework.* O'Reilly Media, Inc., 2010.

[2] Emotiv, *Emotiv Software Development Kit*, 1st ed.

[3] S. Livingstone and S. Fraser, *Beginning C# XML: Essential XML Skills for C# Programmers.* Wrox Press Ltd., 2002.

[4] G. Inc, "Visualization: Annotated time line," Website, http://code.google.com/apis/chart/interactive/docs/gallery/annotatedtimeline.html.

[5] A. Freeman, "Windows presentation foundation," *Introducing Visual C# 2010*, 2010.

[6] carlosga, "Chronos wpf," Website, http://chronoswpf.codeplex.com/.

[7] E. Gamma, *Design patterns: elements of reusable object-oriented software.* Addison-Wesley Professional, 1995.