Methodology & Requirements Document

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

7th January 2012

Swansea University
Prifysgol Abertawe

Contents

1 Introduction

The aim of this document is to introduce the project, we will be undertaking over this 8 month period. To achieve this, the document has been split into 2 sections, beginning with the methodology and ending with the requirements.

In the the Methodology section, we provide an in depth narrative description of the project, paying particular attention to describing the task that the project is attempting to solve. The aim of this narrative description is to give the reader context for the remainder of the document. Additionally, this section includes important project details, such as - Client information, development methodology and an in-depth risk analysis.

In the Requirements section, we will provide a formal list of both the functional and non-functional requirements that the project must conform to. The requirements section is intended to provide a list of functionality and attributes that the project is expected to attain, it does not however give any hints as to how the project should be implemented. For this detail please see the Specification document.

2 The Client

This project is being fulfilled on the behalf of Pingar™ . Pingar provide enterprise level solutions for indexing and searching unstructured data that most companies have sitting in their systems, without an obvious means of using the data.

The company has bases throughout the world, with the headquarters being based in Auckland, New Zealand and other sites in London, Hong Kong, Banglore, and Singapore. One subsidiary company of Pingar, Kaimai Research, is based on the Swansea University campus, within the Digital Technium Building. As such, we have a local contact within this company - John Beer. John Beer is one of the co-founders of Pingar itself, and the strategic research advisor for Kaimai Research. John Beers contact details are as below:

Kaimai Research Ltd,
Digital Technium Swansea University,
Singleton Park,
Swansea,
SA2 8PP
Email: john.beer@pingar.com

However, our project is designed to aid in the development of new research projects within the company. As such, we report to the Chief Research Officer at Pingar - Dr Alyona Medelyan who is based in the New Zealand office:

Pingar
The Smart Business Centre
65 Chapel Street
Tauranga 3110
New Zealand
Email: alyona.medelyan@pingar.com

Additionally, during this project we consult with Dr Anna Divoli (anna.divoli@pingar.com) who is the senior software researcher at Pingar (based in the NZ headquarters), and will also be one of the end users of the finished project.

3 Project Introduction

We devote this section to explaining what it is the project should allow the Client to achieve and the problem that it should help to solve.

The original brief of the project was to design a tool that allowed researchers at Pingar to visually quantify and compare user interfaces against one another. The twist to this would be that the project must make use of the Emotiv EEG brain scanner as a metric in the comparison. The Emotiv device is a commercially available, 16 channel EEG scanner that is both user friendly and ergonomic. The device provides a new and interesting source of data that will hopefully complement existing techniques for evaluating user interfaces. An EEG scanner is a device that measures the electrical activity on the surface of the skull. The micro volt changes detected by the device relate directly to cognitive activity as it occurs within the users brain.

Since the project is being fulfilled for the research department of Pingar, the idea is that the project would be used to compare the user interfaces of two or more prototypes against one another, and return a useful metric for evaluation. The typical way of conducting such an evaluation would be through performing a user study and then drawing conclusions from the results of that study. The project should aid this user study through collecting additional metrics and recording the users interaction with the prototypes, allowing researchers to review the study at a later date.

These prototypes are high fidelity, functioning web pages that are typically (but not always) search user interfaces. As such the project will need to record the following:

- The users interactions with the web page.
- The users “brain activity”

We class this part of the project as the “Web Browser” , since the users interaction will occur within a web browser (since they are viewing standard web pages). However, the function of this web browser will be greater than the standard web browser, since it is expected to collect and store these additional data points.

Having collected this data we need a simple and intuitive way of presenting the collected information to the researcher. Since much of the data (especially that from the brain scanner) is a stream of numerical data - it is mostly incomprehensible. As such, we need some appropriate visualisation to present the findings from the recordings.

The visualisation is expected to aid researcher at Pingar in the discovery of interesting data from the recordings. For example, should one interface cause additional cognitive load over another interface, then this additional load should be clear for the researchers to see on the visualisation, allowing them to conclude that interface A should be considered over interface B since it requires less cognitive process (implying that the interface is conceptually easier to grasp and that the user is required to "think less" in general).

We remember that the primary objective of the project is to compare two or more user interfaces. As such the visualisation must support multiple recordings. With this functionality, the researchers at Pingar will be able to visually compare prototype interfaces side by side and perform quantitative analysis using this project.

To conclude and summarise this section, the aim of this project is to provide a 2 stage solution, that will aid researchers in comparing web based user interfaces. In Stage 1, the project should automatically collect the data from a users browsing session. In Stage 2, the project should provide a interface to the collected data, allowing researchers to compare the prototype user interfaces. From this, the researchers should be aided in deciding which interface they wish to pursue and improve upon - again based on the findings from this project.

4 Methodologies

In this section we will look at the chosen methodology for this project and discuss the reasons for deciding upon this methodology. Additionally, we will look at some of the rejected methodologies and discuss our reasons for deciding against that approach.

A software development methodology can be thought of as a framework that provides a project with a general structure and a cycle of development. A good methodology will enforce certain qualities upon a project, and ensure a quality end result. Not all methodologies are suited to certain projects, as such the choice of the "correct" methodology is critical.

4.1 Chosen Methodology - Prototyping

For this project we have opted to follow the Prototyping Methodology. The prototyping methodology can be summarised in a single sentence as the following:

"Prototyping is the process of developing small components of the overall system, quickly, and reviewing these prototypes with the customer, to gain qualitative feedback early in the development cycle"

Before we look at how this methodology is appropriate for this project, let us first look at the steps involved in applying the Prototyping methodology. Below is an outline of the steps involved:

1. *Identify Basic Requirements* - This initial step is spent developing an understanding of what it is the client would like us to achieve in the course of this project. The emphasis is placed on "basic" requirements since the methodology presumes that the clients themselves are not 100% sure about what it is they wish to achieve. The methodology aims to derive these requirements through the development and subsequent review of prototypes.

2. *Develop Initial Prototype* - The initial prototype provides the customer with an initial direction in which the system is currently progressing. The prototype is mostly a demonstration of the front-end (User interface) since the clients are typically non-technical. This makes for a very quick turn around on the this initial prototype. It also gives a developer a good idea as to the technical requirements of a project, allowing them to prepare for further development.

3. *Review* - The initial prototype is evaluated by the client and revisions/ improvements are discussed with the developer. It is during this stage that the client "realises" what it is that they actually want or need from the project. It is important that a variety of users are present during this review, in order to gain the perspective of all the end users of the system.

4. *Revise and Enhance* - During this stage the developer, using the comments received in the previous stage, enhances the previous prototype to include the additional functionality. Stages 3 and 4 are then repeated upon, with incremental improvements and refinements, based on the input of the client, until finally reaching the finished project.

There are numerous traits of the prototype methodology that make it well suited to this project. The first of these traits are the methodologies extensive integration of the end-user into the development cycle. Since we are developing a tool that helps researchers perform qualitative analysis upon a user interface, it is essential for us to include these researchers into the development process. Doing so, ensures that the system we develop is relevant to the companies needs. The methodology includes the end-user in stages 2 and 4, where their input goes towards the development of the next iteration of the project.

The second trait is the fact that we are developing this project for a client. The prototype methodology is a ideal methodology when you are developing something for someone else (i.e. not an internal project). This is the case, since the methodology allows the true functionality to "unfold" as development progresses. This may seem incorrect and some would argue that functionality requirements should be established prior to beginning the development stage. However it is common that clients are either uncertain of their own requirements, or that they introduce changes at the development stage anyway. By

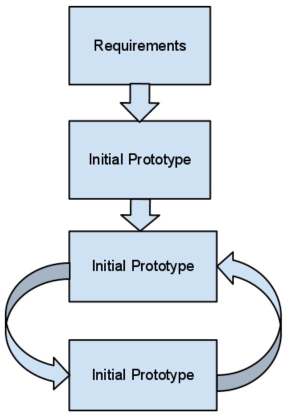

Figure 1: A diagrammatic structure of the Prototyping Cycle.

building this fact into the development cycle, we can plan and accommodate these additions without rescheduling. This also gives the project the benefit of having a realistic time schedule.

The third attribute, is the methodologies simplicity. Although possibly a minor point, having a simple methodology with a linear focus, provides the developer with a single goal and little overhead to concern himself with. This may not be the case with other methodologies which impose many parallel actions that overlap and burden the developer, which can result in a failing project.

Another trait of the methodology is the fact that it is not designed for just a team based development. It can certainly be applied by a team, but it does not explicitly require a team. This means we can use the methodology without modifying it.

Finally, since the project we are developing is in many ways “new ground”, it is important for us as the developer to have a sound understanding of the technical requirements for developing this project. The prototype based development cycle will give us an early indication of the technical requirements and allow us to adjust our time-plan in accordance to these findings.

If we were to criticise this methodology in respect to this project we would draw on the following conclusions.

One problem with having many prototypes when developing a system can be the potential of confusing the client. With many different and incremental prototypes, it is understandable that a client may become overwhelmed and confused with the process. We aim to reduce this factor by clearly communicating each prototype and utilising a clear revision scheme.

Another problem can be the fact that too much time can be spent developing the prototype resulting in missed deadlines or an incomplete system. This fact is also relevant when refining or enhancing prototypes, where “feature creep” from the client becomes an issue. We believe that most methodologies are prone to this type of problem, and it requires diligence on the developers part to ensure that this does not occur.

One final problem attributed to this approach is that a developer may become attached to a particular prototype. If this should occur then the developer will likely develop a system he/she wants instead of refining or scrapping the prototype altogether. The implications of this are obvious, and it is likely that the project will fail. We aim to reduce this burden by conducting regular meetings between the developer and his supervisor, and the client.

4.2 Other Methodologies

Having decided upon implementing the project using the Prototyping methodology, lets have a brief analysis of the other methodologies available to us. We provide some brief reasoning as to why we ultimately decided against using these methodologies.

4.2.1 The Waterfall Model

The waterfall model is a linear model which follows a rigid set of stages, non of which are repeated or revisited. The steps go as such:

1. Requirements
2. Design
3. Implementation
4. Integration
5. Validation
6. Installation
7. Maintenance

Despite the model being fairly well established, we definitely see very little relevance of the model to this project. The primary problem with this approach is that it presumes that all parties have perfect knowledge. We see that requirements are derived immediately and are rigid from then on, leaving no room for modification in latter parts of the project. This is clearly a major flaw in this project, as we are developing an entirely new type of software application, the client is not going to have a definitive and rigid set of requirements from the get go.

The second flaw with this model, is that it does not include the end user throughout the development process. We see that after the initial requirements briefing, no additional user input is considered. For the reason described above, this will likely lead to an application that does not fulfill the users needs, or is not relevant at all.

Finally, we feel that the model is too rigid for this project. We feel the Prototyping approach we have chosen gives much better flexibility in our decision making, whilst maintaining some structure. We feel that the Waterfall model is to idealistic for this type of non-trivial project, and as such we will not be pursuing it.

4.2.2 The Spiral Model

The spiral model is an iterative adaption of the Waterfall model, described above. The model is significantly more complex than the Waterfall model, combining 4 base stages :

1. Determine Objectives
2. Identify and Resolve Risks
3. Development and Test
4. Plan the next iteration

However each stage contains a number of sub-stages, and as we see from stage 4 - the model is iterative meaning many cycles will typically be completed during the course of the development.

Despite this additional complexity of the model, the introduction of iteration over the model makes it much more appealing to this project. We see that in many ways, the Spiral model is in fact a well specified form of the Prototyping methodology. We, gather requirements (1), we develop prototypes(3) and then meet with the client and repeat the process (4). There are however some differences.

To begin with the model has been designed almost exclusively for large projects with (typically) large teams. This is evident when each iteration is designed to take between 6 months and 2 years. So the scope of this model is far to broad for this comparatively small project, and as such would need adapting to fit this fact.

The second problem with the spiral model is that it also suffers from being a rigid model, just like the waterfall model. Unlike the Prototyping methodology, you cannot quickly develop a prototype, show the client, refine, show the client ... , this rapid development is burdened by the requirement to test and verify the prototype which somewhat misses the point of a prototype. Since each stage of the model is well defined and rigid in some senses, it cannot be applied to a project that requires rapid development and refinement typically on a weekly basis.

4.2.3 Agile Method

The Agile development methodology is both an iterative and incremental methodology. The methodology is a comparatively recent approach, meaning it is suited to some of the modern day challenges that face software developers.

The Agile method consists of many small iterations that typically involve a team working through a full "Software development cycle" (typically a waterfall model). The idea is that at the end of each iteration a working piece of software exists, with minimal functionality that incrementally increases with each iteration.

In many senses the Agile method is fairly similar to the chosen methodology, but does contain slightly more structure by including traditional development practices. It was not chosen for this project however since we felt that the requirements of fulfilling a full software development cycle on each iteration was too much of an overhead for our small project. It is clear that the Agile method is designed with a team in mind.

One example of a Agile method is the Scrum method. The method is a group centric development methodology which imposes many of the agile methods described above. The method was not considered for this project since the method is not adaptable to a single developer. The method consists of 3 fundamental roles, which cannot (trivially) be converted to single developer project, as such we did not pursue this methodology.

5 Risk Analysis

We have dedicated the following section to analysing the potential risk that may be encountered during the course of this project. The identified risks are presented in tabular format for easy identification and consumption. We have used a custom scoring system to weigh each risk relatively. Below is a brief explanation of this system (identified by columns 5 - 8 in the table).

Likelihood The likelihood of a risk occurring. This is the measure of how likely we believe a risk is of occurring during the development of this project. This is measured on a scale of 0 - 10, with

- 0 - Not at all likely (Almost definitely not going to occur)
- 10 - Very likely (Almost definite)

Severity The impact that the occurrence of that risk would have on the overall project. This is a measure of just how much of a threat a particular risk is to disrupting the progress of the project. This is measured on a scale of 0 - 10, with

- 0 - No Threat (Has no impact upon the project)
- 10 - Significant Threat (Has a significant impact upon the project)

Level Of Control The measure of our ability to control or alleviate this risk and its potential threat. This is measured on a scale of 0 - 10, with

- 0 - No Control (Factors over which we have no control e.g. Illness)

- 10 - Full Control (Factors which we have complete control e.g. Communication)

As mentioned above we developed this ranking system to be relative. This means we can look at all of the identified risks and look at a single indicator and get a relative metric of the danger of that particular risk. We have called this factor - *Significance*. This is calculated according to the following formula:

(Likelihood + Severity) - Level Of Control = Significance

Using this single metric, we can trivially compare each risk against another. Significance is measured on a scale of 0 - 20, with with

- 0 - No significance (This risk can almost be disregarded)
- 20 - Very Significant (This risk is a combination of great severity and an almost guarantee of occurring)

Below is the table of the risks we have identified in this project.

Risk Id	Project Area	Title	Description	Likelihood	Severity	Level of Control	Significance	Risk Strategy	Risk Mitigation
RISK1	General	Short term illness	Should the developer fall ill for a brief period, then he will have limited contact with the project during that period.	4	7	0	11	Careful Time Planning	We hope to overcome any short term illness by adding contingency periods to the plan.
RISK2	General	Long Term Illness	Should the developer fall into long term illness, he will not be able to contribute to the project..	2	10	0	12	None	In this worst case scenario, the project would have to paused indefinitely until the developer has recovered.
RISK3	General	Poor time management	If the developer fails to manage his time properly then it is likely that the project will overrun.	4	7	7	6	Regular progress meetings	By conducting regular meetings between the developer and his supervisor and clients, the project should remain focused and be delivered on time.

continued on next page

RISK4	General	Avalanche Effect of missed deadlines	A delay in one task may cause further delays in other tasks which rely on it	7	7	7	7	Planning	This risk can be mitigated by performing careful planning followed by regular reviews to keep track of progress.
RISK5	Personal	High coursework load	The developer may have periods of high coursework load, meaning his attention to the project may be reduced.	7	6	7	6	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.
RISK6	General	Company Loses Interest	Should the company lose its interest in the project, it would leave the developer with very little motivation to continue.	3	8	9	2	Communication with the client.	Keeping the client informed with regular meetings will ensure that they maintain their interest.
RISK7	General	Company Demands	Should the company expect or demand too much of the developers resources, then it is likely that the project will overrun.	5	6	8	3	Communication with Client	By informing and updating the client to the deadlines faced by the developer, we can hopefully gain an understanding between client and developer.

continued on next page

RISK8	Judgement	Failure to judge time and resources.	Should the developer fall to judge the projects requirements in terms of time and complexity.	4	7	7	4	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.
RISK9	Specification	Significant changes of client requirements	Should the client have a continually changing specification, then the project will become unfocused and little progression will be made.	6	6	6	6	Communication with Client	This risk should be mitigated through clear communication between the developer and the client. The client should be made aware of the necessary time constraints when appropriate.

continued on next page

RISK10	Requirements	Poor understanding of client requirements	The developer may fail to correctly understand all of the client requirements at some level.	5	7	7	5	Communication with client	This risk can be mitigated through comprehensive communication with the client. Additionally this risk should be reduced through our chosen design methodology.
RISK11	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 sufficently experienced with these technologies.	5	4	8	1	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 developer sufficent notice of what his technical knowledge needs to be.

continued on next page

RISK12	Hardware	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	7	4	7	Taking care with the device.	The headset is designed for gaming, so is somewhat ruggedized but care must still be taken. Additionally the department has an additioanl headset which can be considered a backup.

Table 1: This table holds the risk analysis for this project.

6 Requirements

In this requirements section we will list all the expected requirements that this project should fulfill. We have gathered these requirements by performing numerous HCI based methods, such as Scenarios and Personas. To see the outcome of this analysis, see the Appendices section.

Term Definition

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.

Personas

Person	Responsibility / Relation to project
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 it Pingar.

6.1 Functional Requirements

In this section we will detail the functional requirements of this project. We have divided this section into two smaller subsections in order to contain relevant details into respective parts. These parts are discussed in the Methodology Document.

6.1.1 Web Browser

Requirement ID	**Description**
WBFREQ1	The Web Browser must allow the Conductor to specify necessary properties for a given experiment, prior to beginning the experiment. These include: • User Details (Name, Email, DOB) • Experiment • Name • Condition • Task • InitialURL
WBFREQ2	The Web Browser must maintain and present a list of previously used properties to the Conductor to aid reuse or varying past experiment properties.
WBFREQ3	The Web Browser must allow for the Conductor to specify the appropriate collection sources. Example of such sources are: • Brain Scanner • Audio • Web Browsing Events e.g. Form submissions • Mouse Location • Snapshots of the Interface interaction The browser is expected to be able to collect data from each of these sources.

continued on next page

continued from previous page

Requirement ID	**Description**
WBFREQ4	The Web Browser must allow the conductor to alter Experiment associated data. This means being able to rename and reassign, Participants, Conditions and Task. There must also be the functionality to fully remove each of these properties.
WBFREQ5	The Web Browser must allow for the collection sources to be calibrated appropriately.
WBFREQ6	The Web Browser must contain a common set of UI features that facilitate the traversal of web based documents e.g. Address Bar, Back and Forward buttons, etc ...
WBFREQ7	The Web Browser must display standards compliant web based documents e.g. HTML, XHTML, XML, etc ...
WBFREQ8	The Web Browser must capture and store the collection sources as specified by the Conductor prior to beginning the experiment.
WBFREQ9	The Web Browser must display the status of the collection devices where appropriate e.g. Sensor status on the brain scanner.
WBFREQ10	The Web Browser must allow the Conductor to safely end the experiment.

6.1.2 Visualisation

Requirement ID	**Description**
VSFREQ1	The Visualiser must allow the Researcher to add records to be visualised. These records must be specified by - Experiment/User/Condition/Task.
VSFREQ2	The Researcher must be able to stack visualisations on top of one another for easy comparison.
VSFREQ3	The Visualiser must display the Brain Scanner data. This is considered the main visualisation, and the displayed features should be selectable by the Researcher. The visualisation should be intuitive and convey the brain data in a non trivial manner.
VSFREQ4	The Visualiser must display the events that occurred in the experiment, in a way that correlates to the main visualisation.

continued on next page

continued from previous page

Requirement ID	**Description**
VSFREQ5	The Visualiser must allow the Researcher to scroll through the visualised temporal and spacial data.
VSFREQ6	The Visualiser must present easily identifiable correlations between the stacked experiments to the Researcher.
VSFREQ7	The Visualiser must display a correlated and real time display of the Document in relation to the Researchers interaction with the visualisation. Example - If the Researcher is inspecting the visualisation at time X, then the Visualiser must show the snapshot of the document at (approximately) time X. (This is only true when the screen capture plug-in is enabled).
VSFREQ8	The Visualiser must allow the Researcher to add annotations to the data in the form of events. These events must then be retained.
VSFREQ9	The Visualiser must allow the Researcher to extract desirable signal values at certain, specified points in time.

6.2 Non-functional Requirements

In this section we conclude this requirements document by looking at the non-functional requirements of this project. We have again split this section into respective subsections.

6.2.1 Web Browser

Requirement ID	**Description**
WBNFREQ1	The Web Browser must allow the User to view the document without interruptions or distractions.
WBNFREQ2	The Web Browser must be intuitive for both the User and Conductor to use.
WBNFREQ3	The Web Browser must be able to collect from many data sources simultaneously.
WBNFREQ4	The Web Browser must be thoroughly documented.
WBNFREQ5	The Web Browser will require a modern Windows 7 64bit based machine with the specification : • Intel Core i5 • 4GB RAM • 100MB of Hard Disk Space • A dedicated graphics card with $>$ 256mb RAM • A recording device - Microphone • A Monitor ($>$19' and minimum resolution - 1280 x 720) Software: • .Net Framework Version 4 • Emotiv Research Edition SDK • Windows SAPI V5 (or grater)
WBNFREQ6	The Web Browser must be extensible.

6.2.2 Visualisation

Requirement ID	**Description**
VSNFREQ1	The Visualiser must be intuitive to use.
VSNFREQ2	The generated visualisations must be simple to grasp and informative.
VSNFREQ3	The Visualiser must be thoroughly documented.

continued on next page

continued from previous page

Requirement ID	**Description**
VSNFREQ4	The Visualiser will require a modern Windows 7 64bit based machine with the specification : • Intel Core i5 • 4GB RAM • 100MB of Hard Disk Space • A dedicated graphics card with > 256mb RAM • A recording device - Microphone • A Monitor (>19' and minimum resolution - 1280 x 1080) Software: • .Net Framework Version 4 • Windows SAPI V5 (or grater)
VSNFREQ5	The Visualiser must be able to handle vast amounts of data, efficiently.

7 Time plan

Below is a look at the various plans we have made with regards to the project. We have planned various aspects of the project accordingly. We have used Gantt charts to present each time plan. Each chart is associated with a particular aspect of the project. Additionally, chart is supported by its own table containing the task and dates.

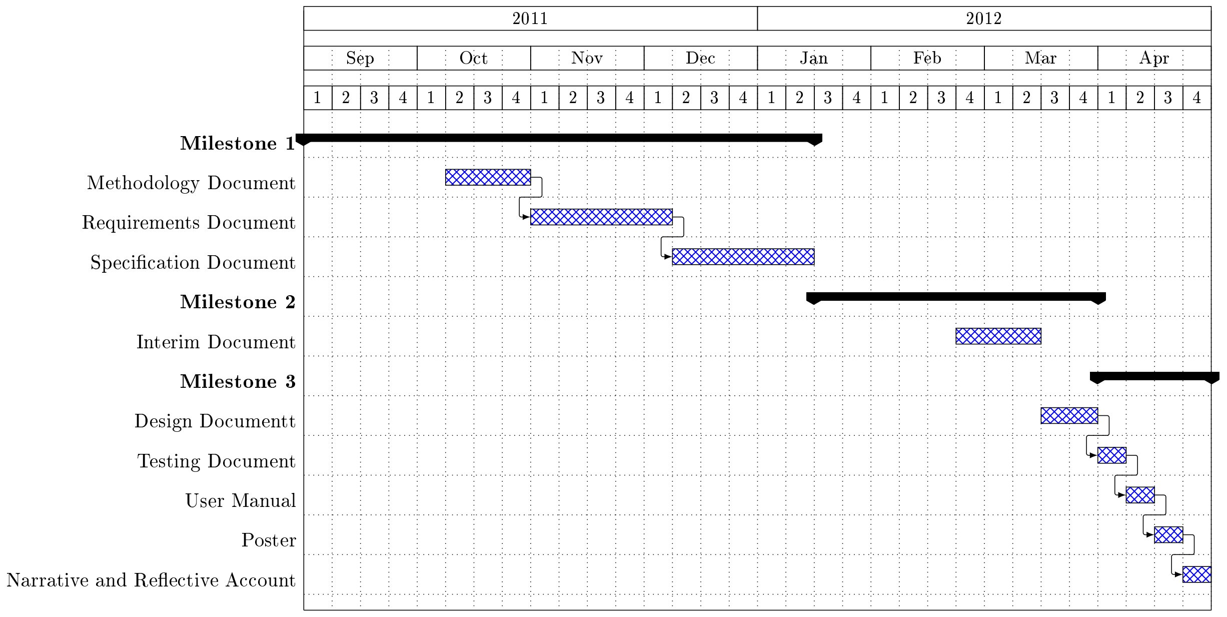

Figure 2: A Gantt chart of the project deadlines.

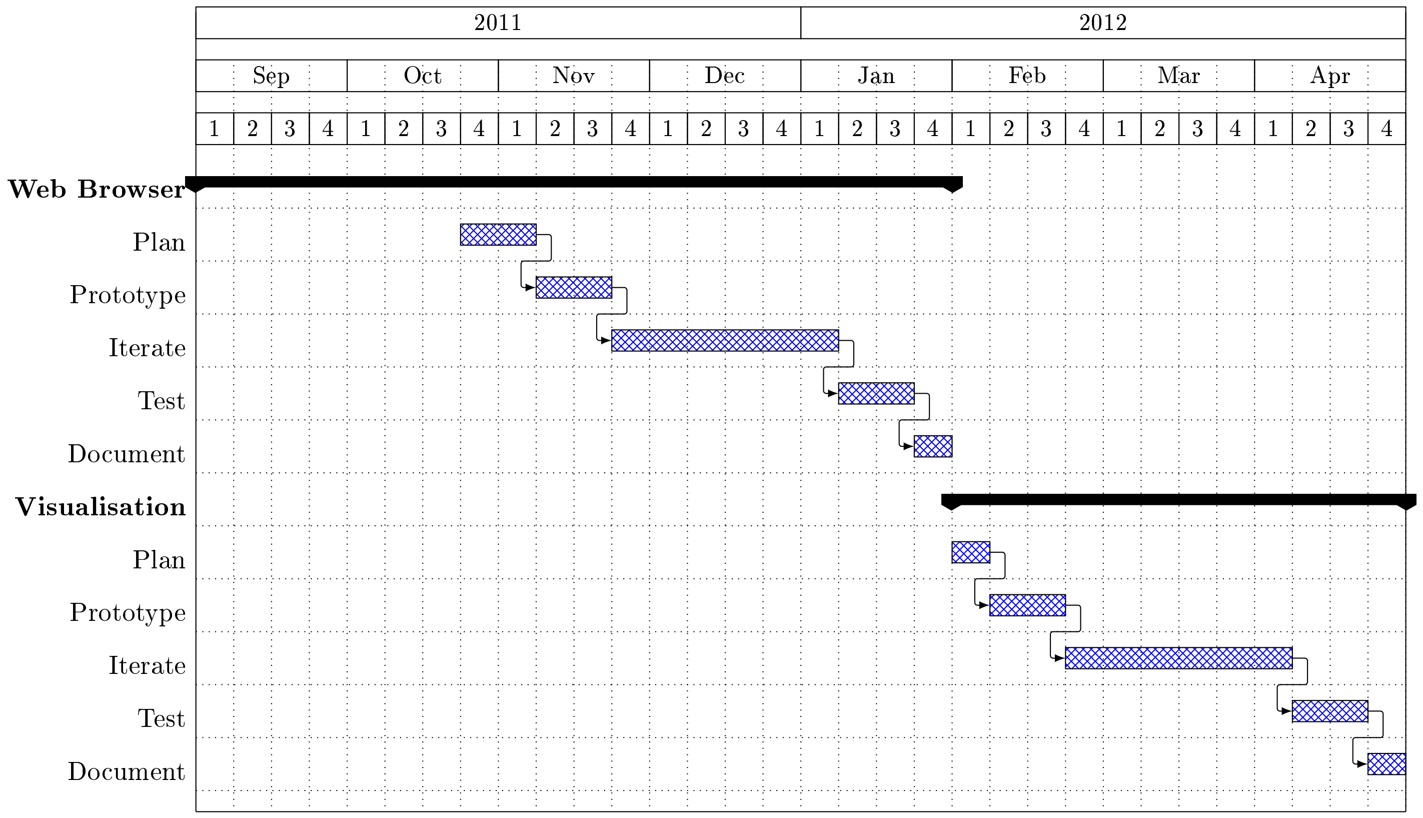

Figure 3: A Gantt chart of the development time plan.

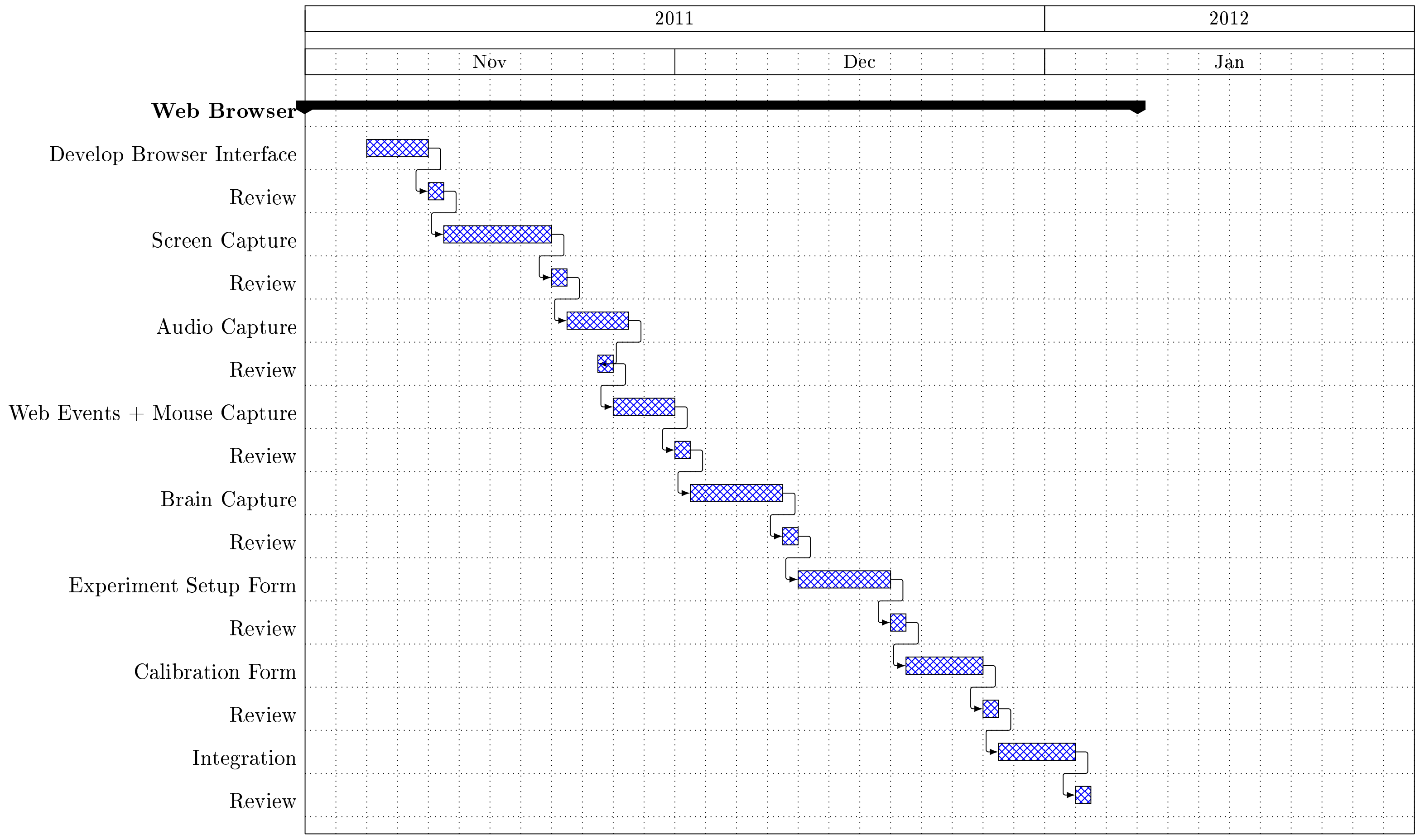

Figure 4: A Gantt chart of the development time plan for the Web Browser.

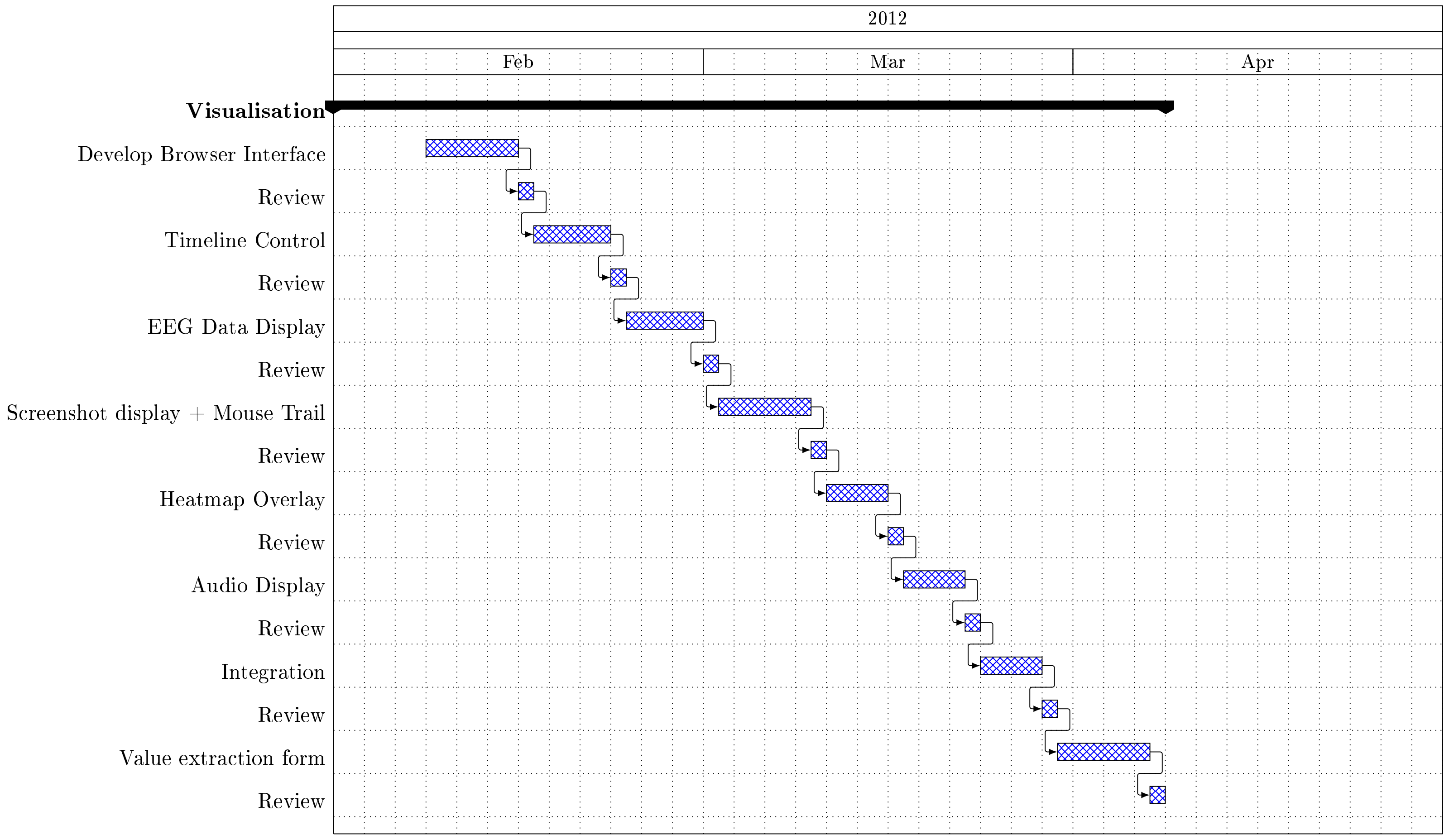

Figure 5: A Gantt chart of the development time plan for the Visualisation.

Task	Start Date	End Date
Milestone 1	9 Sept 2011	09 Jan 2012
Methodology Document	7 Oct 2011	28 Oct 2011
Requirements Document	1 Nov 2011	7 Dec 2011
Specification Document	9 Dec 2011	7 Jan 2012
Milestone 2	09 Jan 2012	30 Mar 2012
Interim Report	20 Feb 2012	01 Mar 2012
Milestone 3	30 Mar 2012	27 April 2012
Poster Presentation	17 April 2012	24 April 2012
User Manual	10 April 2012	17 April 2012
Design Document	13 Mar 2012	28 Mar 2012
Testing Document	01 Apr 2012	08 April 2012
Narrative and reflective account	24 April 2012	26 April 2012

Table 2: This time plan for the document submission aspect of the project.

Task	Start Date	End Date
Web Browser	09 Sept 2011	09 Jan 2012
Plan	23 Oct 2011	07 Nov 2011
Prototype	08 Nov 2011	22 Nov 2011
Iterate	23 Nov 2011	07 Jan 2012
Test	08 Jan 2012	22 Jan 2012
Document	23 Jan 2012	01 Feb 2012
Visualisation	1 Feb 2012	27 April 2012
Plan	02 Feb 2012	09 Feb 2012
Prototype	10 Feb 2012	22 Feb 2012
Iterate	23 Feb 2012	07 Apr 2012
Test	08 Apr 2012	20 Apr 2012
Document	21 Apr 2012	25 Apr 2012

Table 3: This time plan for the development aspect of the project.

Task	Start Date	End Date
Develop Web Browser Interface and implement basic functionality	7 Nov 2011	11 Nov 2011
Review Prototype with Client	11 Nov 2011	11 Nov 2011
Develop Screen Capture Functionality + Implement Comments from review	11 Nov 2011	18 Nov 2011
Review Prototype with Client	18 Nov 2011	18 Nov 2011
Develop Audio Capture + Implement Comments from review	18 Nov 2011	23 Nov 2011
Review Prototype with Client	23 Nov 2011	23 Nov 2011
Develop Web Events + Mouse Capture + Implement Comments from review	23 Nov 2011	30 Nov 2011
Review Prototype with Client	30 Nov 2011	30 Nov 2011
Develop Brain Capture + Implement Comments from review	30 Nov 2011	10 Dec 2011
Review Prototype with Client	10 Dec 2011	10 Dec 2011
Develop Setup Form and Tie to DB + Implement Comments from review1	10 Dec 2011	15 Dec 2011
Review Prototype with Client	15 Dec 2011	15 Dec 2011
Develop Calibration Interface + Implement Comments from review1	15 Dec 2011	20 Dec 2011
Review Prototype with Client	20 Dec 2011	20 Dec 2011
Tie all components together (Integration) + Implement Comments from review1	28 Dec 2011	7 Jan 2012
Present Project to client	7 Jan 2012	7 Jan 2012

Table 4: This time plan for the development of the Web Browser Components.

Task	**Start Date**	**End Date**
Develop Main Interface and implement basic functionality	10 Feb 2012	20 Feb 2012
Review Prototype with Client	20 Feb 2012	20 Feb 2012
Develop Basic Timeline Control + Implement Comments from review	20 Feb 2012	27 Feb 2012
Review Prototype with Client	27 Feb 2012	27 Feb 2012
Develop EEG Data Display + Implement Comments from review	27 Feb 2012	5 Mar 2012
Review Prototype with Client	5 Mar 2012	5 Mar 2012
Develop Screenshot display with Mouse Trail Overlay + Implement Comments from review	5 Mar 2012	15 Mar 2012
Review Prototype with Client	15 Mar 2012	15 Mar 2012
Develop Heatmap Overlay + Implement Comments from review	15 Mar 2012	20 Mar 2012
Review Prototype with Client	20 Mar 2012	20 Mar 2012
Develop Audio Display + Implement Comments from review	20 Mar 2012	25 Mar 2012
Review Prototype with Client	25 Mar 2012	25 Mar 2012
Tie all Components together + Implement Comments from review	25 Mar 2012	2 Apr 2012
Review Prototype with Client	2 Apr 2012	2 Apr 2012
Develop Value extraction form + Implement Comments from review	2 Apr 2012	7 Apr 2012
Present Project to client	7 Apr 2012	7 Apr 2012

Table 5: This time plan for the development of the Visualisation Components.

Appendices

Persona's

Chief Researcher

The chief researcher is the person who commissions and oversees the user study. They see user studies as a tool for validating or rejecting a particular hypothesis they have. General this hypothesis is relative to an up-coming product, but this is not necessarily the case (it could be an analysis of an existing product for example). The chief researcher typically does not conduct the user study, instead she has a member of her team to do this task. The chief researcher is responsible for a reasonably large group of researchers, who are usually equally qualified, but not as experienced as the chief. The chief researcher is at a minimum educated to university standard, typically to a post-doctoral level. Despite her responsibilities as a manager, the chief researcher is still, ultimately answerable to her superiors, who like to see insight rather than theoretical or unproven ideas.

User Study Coordinator

Similarly to the Chief researcher, the user study coordinator is educated to a minimum of university standard, and typically post-doctoral. They are typically researchers themselves, working under the chief researcher. Their duties are typically to work on ideas set out by the chief researchers. The researcher typically employs the User Study method to confirm ultimately a hypothesis held by their superior(s). They are therefore typically responsible for devising the user study (in combination with the chief researcher). Having decided on a study format, the coordinator is then the individual responsible for conducting the study, logging the results, and presenting it to the Chief Researcher.

User Study Participant

A participant in a study can be split into 2 unique categories.

Specialist Participant

- Varying levels of education in the study field
- Motivation is typically split
  - Monetary reward
  - A desire to improve / give back to his/her field

- Have in-depth knowledge of the study contents
- Have little knowledge of the studies intention

A specialised participant is typically recruited and used in user studies when the system under test is an expert or specialist system. Examples of such systems include: medical search systems and medical expert systems . The users motivation for participating in the study is usually dependent on their level of education. A student in the field would typically partake in the study for monetary reward, whereas a mature professional would likely be looking to contribute to his/her field. A general participant has the following attributes:

General Participant

- Varying levels of education in all fields
- Motivation is typically monetary reward
- Has little knowledge of the studies purpose and its contents.

Business Manager

- Is typically removed from the process of the study
- Interested in results not the process of retrieving them
- May not be as educated in the field as the Chief researcher
- Ultimate decision maker

Scenario

A scenario is a HCI technique that translates the process of completing a task into a narrative story. A scenario is therefore an - "Informal narrative description". We have applied this process, based on my meeting(s) with Pingar, and have produced the proceeding scenario. The scenario is from written from the point of view of a conductor of the user study within pingar.

"Beth is a researcher who has developed a new version of an existing project. She wishes to see how this version compares to the previous. She has already conducted, collected and processed the results of a user study on the original version. She wishes to do the same this time, using an identical procedure, but for the new iteration of the project.

She begins by recruiting individuals with specific knowledge in the projects field. It is typical that she reuses participants that were used in the original study. It is important in this project that the users are well versed in the projects field (e.g. bio-medicine, computer science etc), however this is not always the case.

Having recruited sufficiently skilled users, Beth devise's a task for them to perform. This task occurs within a document, typically a website. Beth also produces a list of questions which she will pose to the users during the study. These questions are typically a mix of quantitative and comment based response questions. She will ask questions which will help to prove or disprove a hypothesis, which is ultimately the goal of the user study.

During the study Beth presents the particular task to the subject and lets them begin the task. Through observation, Beth makes handwritten notes on particular decisions and interactions the user makes with the interface, that she deems to be interesting. When appropriate Beth poses the questions to the participant. Despite the list being in a logical and progressive order, often during the study, the need arises to ask questions "out of order" - due to a particular interaction/decision on the users part. The responses are recorded on paper. Additional comments from the user are also recorded. Once the task is complete, and all the questions have been answered, the study is over. This process is repeated for all the users in the study. Beth then begins the significant task of looking through the data.

Beth translates the notes into a easy to manage spreadsheet. She compares each of the data points such as the questions and comments (posed / received) during the study. Beth is looking for common patterns, specifically relating to the hypothesis. Beth's aim is to get a cumulative opinion of the studies results based on the users feedback. Additionally Beth calculates a quantitative value based on the users answer to the quantitative questions.

Finally, Beth manually searches through the data to find evidence to support the hypothesis under analysis."

Hierarchical Task Analysis (HTA)

From the above scenario we can draw the following hierarchy of the tasks.

1. In order to prove/disprove a hypothesis, we conduct a user study
   - (a) Decide on the system under test
     - i. If necessary, recruit sufficiently skilled users
     - ii. Devise a specific task for the user to perform during the study
     - iii. Devise quantitative and descriptive questions for the study
     - iv. Specify a document which the user will attempt the task upon
     - v. Formally log the hypothesis under scrutiny
   - (b) Perform the study
     - i. Present the user to the document and the task in hand
     - ii. Inform the user of what we wish to learn from them and the study
     - iii. Begin the study
       - A. Record user comments
       - B. Pose questions to the user - record response
       - C. Make general notes of the study
       - D. Record impromptu comments from the user.
     - iv. The Study is ended.
   - (c) Prove / Disprove the hypothesis by analysing the data.
     - i. Search through the data looking for patterns that help decide an outcome
     - ii. Compare users against each other
     - iii. Conclude a decision on the hypothesis
     - iv. Compare against old document results (where appropriate)