WEB USER REQUIREMENTS: A SUPPORT FRAMEWORK FOR STUDENTS

Volume 2

Andrew Paul Bingham

A thesis submitted in partial fulfillment of the requirements of Teesside University for the degree of Doctor of Philosophy

July 2012
Contents

Appendix A1 - RUMM Survey Questionnaire (AR1)................................. 259
Appendix A2 - RUMM Fill In Forms....................................................... 261
Appendix A2.1 - DFU Module Specification......................................... 264
Appendix A2.2 - IID Module Specification.......................................... 268
Appendix A2.3 - RUMM Survey Codebook........................................ 273
Appendix A2.4 - Excerpts of Observational Logs (AR1,2 and 3)......... 274
Appendix A2.5 – OBS Module Specification .................................... 277
Appendix A3 – Related Work ............................................................... 283
Appendix A3.1 - SIGSAND Conference Paper................................ 325
Appendix A4 - UKAIS Phd Consortium Conference Paper............... 333
Appendix A5 - Online Survey Questionnaire (AR2).............................. 342
Appendix A6 - Survey Data Table......................................................... 346
Appendix B1 - CAWE Development and Implementation............... 347
Appendix B1.1 - Software Requirements Specification.................... 370
Appendix B1.2 - Student Email for Survey........................................ 374
Appendix C1 - MakeAssociation PHP class....................................... 375
Appendix C2 - requirementDocumentPrint PHP class...................... 383
Appendix C3 - Student Opinion Survey............................................... 403
Appendix C4 - Usage Data Tables....................................................... 428
Appendix C5 - Student Generated SRS Document.............................. 434
Appendix A1 - RUMM Survey Questionnaire

Please fill in the following questionnaire as fully and honestly as you can, your response and comments will help determine where the next stage of the research will go. Thank you.

1. How useful did you find using this approach was in helping you define your audience?

<table>
<thead>
<tr>
<th>Very Useful/helpful</th>
<th>Useful/helpful</th>
<th>Neither helpful nor unhelpful</th>
<th>A little unhelpful/confusing</th>
<th>Complicated &amp; very unhelpful</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Please tick a box

2. If you have used other user defining approaches in the past how do you think this approach compares?

<table>
<thead>
<tr>
<th>Never used other approaches</th>
<th>Much more useful</th>
<th>More useful</th>
<th>Not much better really</th>
<th>I’ll stick with my original approach!</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Please tick a box

3. Do you think there are any points missing from the list that should be considered?

Yes  No  If “Yes” Please state what point(s):

If “Yes” Please state what point(s):
4. Do you think there is any unnecessary information being gathered through this method?

- [ ] Yes
- [ ] No

If “Yes” Please state what information:

5. The next stage of the model will be to link the user model you have created with issues that you need to consider in your design. It is anticipated that this would be generated online and would provide a list of design guidelines that is UNIQUE to the user model you have created. If such a tool were available to you how useful and helpful do you think it would be to you as a designer?

<table>
<thead>
<tr>
<th>Very useful &amp; helpful</th>
<th>Useful and helpful</th>
<th>Neither helpful nor unhelpful</th>
<th>Unhelpful</th>
<th>Very unhelpful</th>
</tr>
</thead>
</table>

Please tick a box
## Appendix A2 - Rapid User Modelling Method (RUMM) Fill In Forms

<table>
<thead>
<tr>
<th>Who is the user?</th>
<th>Primary Users</th>
<th>Secondary Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Children – KS 0 (0-4 yrs)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Children – KS 1 (4-7 yrs)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Children – KS 2 (7-11 yrs)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Children – KS 3 (11-14 yrs)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Children – KS 4 (14-16 yrs)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adults (17-30)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adults (30-50)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adults (50+)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gender &amp; Culture</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Male</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Female</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Both</td>
<td></td>
<td></td>
</tr>
<tr>
<td>General Cultural Background (?)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Level of computer use/competence</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Novice</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intermediate</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expert</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Employment job/type (?)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What does the user expect to do with the application?

<table>
<thead>
<tr>
<th>Use it as the main part of their job?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assist them to do their job?</td>
</tr>
<tr>
<td>Allow them to buy something online?</td>
</tr>
<tr>
<td>Assist them to find out information - specific and/or general?</td>
</tr>
<tr>
<td>Provide fun or leisure activities?</td>
</tr>
<tr>
<td>Help them learn something?</td>
</tr>
</tbody>
</table>

When will the user use the application?

<table>
<thead>
<tr>
<th>In the course of their job?</th>
</tr>
</thead>
<tbody>
<tr>
<td>At home in their own leisure time?</td>
</tr>
<tr>
<td>In library or other public access point?</td>
</tr>
</tbody>
</table>

How will the user use the application?

<table>
<thead>
<tr>
<th>At work</th>
</tr>
</thead>
<tbody>
<tr>
<td>With modem link (what speed)</td>
</tr>
<tr>
<td>With broadband connection</td>
</tr>
<tr>
<td>Stand alone CD/DVD</td>
</tr>
<tr>
<td>At home</td>
</tr>
<tr>
<td>With modem link</td>
</tr>
<tr>
<td>With broadband connection</td>
</tr>
<tr>
<td>Stand alone CD/DVD</td>
</tr>
<tr>
<td>On PC (specification)</td>
</tr>
<tr>
<td>On Mac (specification)</td>
</tr>
<tr>
<td>On Linux platform (specification)</td>
</tr>
<tr>
<td>Kiosk</td>
</tr>
<tr>
<td>PDA</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>With assistive technologies? (specify)</td>
</tr>
</tbody>
</table>

Now write a brief summary of your primary users:

Now write a brief summary of your secondary users:

Now identify some of the implications of these issues for your design under the following headings:

- **Layout**
- **Colour**
- **Content**
- **Navigation**
### Module Title
Design for Usability

<table>
<thead>
<tr>
<th><strong>Module Status</strong></th>
<th>New</th>
<th><strong>Date of Official Approval</strong></th>
<th>2003</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Module Title</strong></td>
<td></td>
<td><strong>Module Status</strong></td>
<td></td>
</tr>
<tr>
<td><strong>SITS Module Code</strong></td>
<td>MUL2003-N</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Module Descriptor</strong></td>
<td>The module aims to give the students an understanding of the design of usable yet creative interfaces. Much of the focus of the module is on web based applications, but many of the approaches and techniques discussed could be applied to the development of other computer interfaces. The module emphasises the importance of the relationship between the user and the system. Key issues include the understanding of human cognitive abilities - memory, vision, problem-solving and reasoning - then builds upon this to consider interaction methods, Usability Engineering Life Cycle, guidelines for 'good' and creative design, and evaluation of interfaces. There is a strong element of practical work in prototyping an interactive interface, including user testing and evaluation.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Key Words</strong> (10 max)</td>
<td>User Centred Design, human cognition, perception &amp; creativity, interactivity, interface development</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Module Type</strong></td>
<td>Standard</td>
<td><strong>Credits (UoT CAMS)</strong></td>
<td>20</td>
</tr>
<tr>
<td><strong>Total Learning Hours</strong></td>
<td>200</td>
<td><strong>Level</strong></td>
<td>II</td>
</tr>
<tr>
<td><strong>Summative Assessment</strong></td>
<td>ICA 100%</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Assessment Marking Scheme</strong></td>
<td>2004 UG Module</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Assignment Submission</strong></td>
<td>School of Computing</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Module Aims

1. To provide students with the knowledge needed to understand the role of the human user as a component in a human-computer system.
2. To challenge the students attitudes towards applying creative and aesthetically pleasing designs to interface development.
3. To provide the student with tools and techniques required to analyse, design, develop and evaluate an interactive user interface, with a focus on Usability Engineering techniques and practices including evaluating if the application meets the needs of the target user group.
4. To provide the student with an understanding of a variety of interaction methods and the appropriate methodologies and guidelines for good design and development.
5. To provide the student with the practical experience of using a prototyping tool for interactive user interface product development.

Indicative Content

1. Introduction to human-computer interaction.
2. The User Centred Development Methodology, including establishing user requirements, preparing a mission statement, conceptual design, implementation, testing and evaluation.
3. Understanding the user: cognition [perception, memory, attention, problem-solving and reasoning]; semiotics, society, culture and gender.
4. Producing interactions to enhance the users experience.
5. Creativity and its role in the design of an effective user interface solution.
6. Evaluation and comparison of traditional design methods and strategy.
8. Conventions, constraints and limitations of the implementation process.
9. Analysis, design, development and evaluation of the human-computer interface.
10. Techniques for building 'user friendly' interfaces.
11. Practical interface development, prototyping and testing.

**Learning Strategy**

The overall strategy for teaching and learning is described in the Framework document. Specific features of this module are: Delivery of underlying theoretical concepts of the field of creative yet usable interface development through a series of lectures, with further in-depth exploration by students via tutorial tasks and structured, directed research. Exploration of practical aspects is via laboratory-based activities, including the use of prototyping tools for interface development, and the use of evaluation techniques for refining and user testing of that interface. Much of the module revolves around active participation by the student, and uses a variety of delivery styles supported by a range of resources to encourage participation, learning and reflection. Activities undertaken by students include completion of lecture-related tasks; tutorial work on aspects such as user analysis, production of design documentation and various evaluation techniques and practices; practical work on interface design for prototype development and user testing and evaluation. There is a mix of group and individual activities.

**Learning Outcomes**

On successful completion of this module, the student will be able to:
1. Demonstrate and document an understanding of the role of the human as a component in a human-computer system.
2. Apply an aesthetic and creative solution to the design of an interface.
3. Analyse, design, develop and evaluate an interactive interface.
4. Use an appropriate design methodology and design guidelines in the production of an application.
5. Implement a prototype interface using a specified application development tool.

**Assessment**

**Assessment Strategy**

The formal mechanism for assessing student achievement is via In-Course Assessment. This will take the form of the design and development of a web-based application that should reflect good usability design principles. The students will also have to demonstrate their knowledge of the needs of the user through a form of scholarly discourse. The assessment will measure all the learning outcomes of the module. The School operates a standard procedure for providing (at least) a minimum level of feedback to students.

**Assessment Criteria**

**Indicative Resources**

Purchase
not applicable

Recommended
"The Essential Guide To User Interface Design : An introduction to GUI design principles and techniques" by Wilbert O. Galitz, Wiley Computer, 2002

Journals not applicable

Electronic
Blackboard Virtual Learning Environment.

Accessibility

The School of Computing endeavours to make all of its modules inclusive and does its best to adopt accessible and inclusive practices but we are aware that we cannot anticipate every possible special needs or requirements. There may be elements of this module (resources, assessment, learning and teaching methods, etc) that may present difficulties for students with special needs. You are strongly advised to check the module details carefully and discuss any potential problems with the Special Needs tutor so that your particular needs can be accommodated wherever possible.
# Appendix A2.2 - IID Module Specification

## TEESIDE UNIVERSITY

### MODULE SPECIFICATION

<table>
<thead>
<tr>
<th>Module Title</th>
<th>Integrated Development</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Module Status</strong></td>
<td><strong>New</strong></td>
</tr>
<tr>
<td><strong>SITS Module Code</strong></td>
<td><strong>MUL4002-N</strong></td>
</tr>
</tbody>
</table>

### Module Descriptor

This module will provide the student with skills necessary to manage the multimedia implementation process. The module will take an integrated approach to the use of multimedia development tools. Students will also be provided with concepts and skills for utilising an appropriate scripting language. This module is appropriate to students who already have an understanding of multimedia production and who wish to extend these skills further. The module will be supported by lectures and practical tutorials. Other resources and additional learning opportunities will be presented via Blackboard and alternative web delivery systems. The module will be assessed by an in course assessment which will involve the development of a substantial multimedia product.

### Key Words (10 max)

Multimedia, Implementation, Scripting, Workflow Management

<table>
<thead>
<tr>
<th>Module Type</th>
<th>Credits (UoT CAMS)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Standard</strong></td>
<td><strong>20</strong></td>
</tr>
</tbody>
</table>

| Total Learning Hours | **200** | **Level** | **IV** |

| Summative Assessment | **ICA 100%** |

**Assessment Marking Scheme**

| 2004 PG Module |
Module Aims

The module aims to:
1. Establish the principles of a well managed multimedia implementation process.
2. Introduce key scripting concepts to enable the development of interactive multimedia.
3. Present appropriate web technologies for the use in developing web based multimedia.
4. Give students the opportunity to gain experience in developing multimedia using an industry standard integrated development package.
5. Encourage the student to reflect on the implementation process.

Indicative Content

Introduction to a 'work flow model' for the management of the implementation process. Effective use of Implementation tools, including industry standard Multimedia/Web Authoring tools. Utilisation of Macromedia Studio, (Flash, Dreamweaver and Fireworks). Fundamentals and concepts of scripting for multimedia and the web. - Problem solving and providing efficient solutions using a range of scripting languages, including Actionscript and Javascript. - Working with objects, functions, conditionals and loops. - Data Types, Variables, Assignment, Input and Output. Overview of future technologies and standards, and how these affect the implementation process. Writing XHTML compliant mark-up. Using CSS to efficiently present content for the World Wide Web. Awareness of the need to address accessibility requirements in the implementation of multimedia and web based applications. Introduction to Server-side technologies,
including FTP and client to server communication. Data validation and security issues. The evaluation process and its integration into the product lifecycle.

**Learning Strategy**

The module will be taught by blend of lectures, practical tutorials and online learning. Blackboard will be used to integrate learning objects, where this is deemed appropriate. This will also include online discussion, which should combine to foster a collaborative learning experience. Lectures will be used to develop understanding of the underpinning theory and concepts involved in implementing integrated multimedia. Case studies will be used to illustrate how the theory relates to commercial application implementation. In addition, active lectures, using a variety of techniques, will enable students to investigate key concepts in relation to the tools, within the session itself. Practical tutorials will concentrate on developing key skills within the implementation environment, including graphical asset production, implementation of multimedia/web based objects and access to server side technologies. Self directed learning, taking into account the level of the module, will be a required from the student.

**Learning Outcomes**

**Knowledge & Understanding**

On successful completion of this module, the student will be able to:
1. Appraise different web technologies and apply an appropriate solution to a given scenario.

**Cognitive & Intellectual Skills**

On successful completion of this module, the student will be able to:
1. Critically evaluate their implementation approach.

**Practical & Professional Skills**

On successful completion of this module, the student will be able to:
1. Compose a well formed scripting solution to address the requirements of a design brief.
2. Design and construct a multimedia solution addressing the requirements of a design brief.

**Key Transferable Skills**

On successful completion of this module, the student will be able to:
1. Manage the implementation phase of a multimedia development lifecycle.

**Assessment**

**Assessment Strategy**

The module will be assessed by in-course assessment (ICA): Using a project brief and specification, students will design and develop a multimedia application, along with a linked 'micro web site'. This will incorporate both the theoretical and practical components.
of the module, such as writing functions for the development of interactions and animations and post-implementation evaluation. Clear assessment marking criteria will be presented within the ICA specification. Individual feedback will be given in line with these criteria. The School operates a standard procedure for providing (at least) a minimum level of feedback to students in line with Minimum Standards.

**Assessment Criteria**

Assessment criteria will be provided, related to the set tasks, stating how marks will be allocated.

**Indicative Resources**

**Purchase**

Not Applicable

**Essential**

This module will require access to a lecture theatre and laboratory facilities that support digital presentations, as well as access to web development tools as outlined below.

1. Non web based tools (e.g. Email, FTP, SSH).
2. Dynamic HTML enabled Web Browsers (e.g. Mozilla Firefox, Internet Explorer).
3. Web Site Development tools (e.g. Macromedia Dreamweaver).
4. Multimedia Development tools (e.g. Flash).
5. Asset creation tools (e.g. Adobe Photoshop, Macromedia Fireworks).


**Recommended**


**Journals**

International Journal of Web Engineering and Technology
Interacting with Computers
New Review of Hypermedia and Multimedia

**Electronic**

Blackboard. http://www.blackboard2.tees.ac.uk

**Accessibility**
The School of Computing endeavours to make all of its modules inclusive and does its best to adopt accessible and inclusive practices but we are aware that we cannot anticipate every possible special needs or requirements. There may be elements of this module (resources, assessment, learning and teaching methods, etc) that may present difficulties for students with special needs. You are strongly advised to check the module details carefully and discuss any potential problems with the Special Needs tutor so that your particular needs can be accommodated wherever possible. Further advice is available from the University Student Services staff.
Appendix A2.3 - RUMM Survey Codebook

Question 1. How useful did you find using this approach was in helping you define your audience?

Very Useful/helpful: a
Useful/helpful: b
Neither helpful nor unhelpful: c
A little unhelpful/confusing: d
Complicated & very unhelpful: e

Question 2. If you have used other user defining approaches in the past how do you think this approach compares?

Never used other approaches: a
Much more useful: b
More useful: c
Not much better really: d
I’ll stick with my original approach: e

Question 3. Do you think there are any points missing from the list that should be considered?

Yes: Y
No: N

Question 4. Do you think there is any unnecessary information being gathered through this method?

Yes: Y
No: N

Question 5. The next stage of the model will be to link the user model you have created with issues that you need to consider in your design. It is anticipated that this would be generated online and would provide a list of design guidelines that is UNIQUE to the user model you have created. If such a tool were available to you how useful and helpful do you think it would be to you as a designer?

Very useful & helpful a
Useful and helpful b
Neither helpful nor unhelpful c
Unhelpful d
Very unhelpful e
## Appendix A2.4 - Excerpts of Observational Logs (AR1, 2 and 3)

Sample of Observational Logs. (Recorded in notebook form and significant observations translated).

### AR1 Observations

<table>
<thead>
<tr>
<th>Date</th>
<th>Location</th>
<th>Observational Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oct 2005</td>
<td>Lab</td>
<td>Students appear to be adopting RUMM in their initial planning stages, but quickly becoming bored! Some students asking how to gather information about the user. Students not completing RUMM ‘at one sitting’.</td>
</tr>
<tr>
<td>Nov 2005</td>
<td>Lab</td>
<td>Students generally need a lot more support in using RUMM than had been expected. It is feared that the reasons for this is that I have not provided them with sufficient support material.</td>
</tr>
<tr>
<td>December 2005</td>
<td>Lab</td>
<td>Some students on the masters module IID wish to modify/extend RUMM to suite their own development practices. Unsure if this is valid, but it does raise some questions regarding the limitations of RUMM in its current form.</td>
</tr>
<tr>
<td>Jan 2006</td>
<td>Lab</td>
<td>Some student expressing if they are able to use persona’s to model the user, rather than the built in ‘user characteristics’ of RUMM.</td>
</tr>
<tr>
<td>June 2006</td>
<td>Office</td>
<td>Some students are submitting incomplete requirements within their ICA’s. From assessing their websites, it seems that there is some relationship to their documented requirements. Generally, much more analysis is taking place than in previous years.</td>
</tr>
</tbody>
</table>

### AR2 Observations

<table>
<thead>
<tr>
<th>Date</th>
<th>Location</th>
<th>Observational Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oct 2006</td>
<td>Lab</td>
<td>Students are jumping into different stages in the modified WURF process meta-model. It had been expected that a sequential approach to be taken. This is potentially problematic, as analysis of functional requirements is dependent on previously completed stages such as tasks.</td>
</tr>
<tr>
<td>Date</td>
<td>Location</td>
<td>Notes</td>
</tr>
<tr>
<td>------------</td>
<td>----------</td>
<td>-------</td>
</tr>
<tr>
<td>Oct 2006</td>
<td>Lab</td>
<td>The association system is causing issues for students. Some students are transposing WURF into their own requirements document.</td>
</tr>
<tr>
<td>Nov 2006</td>
<td>Lab</td>
<td>The separation of users into Actors seemed to be working well with students, particularly those developing web applications that consume a web service. Having unlimited users is providing students with much more scope for modelling the whole application, rather than being limited to primary and secondary (RUMM).</td>
</tr>
<tr>
<td>June 2006</td>
<td>Office</td>
<td>Assessment of the student submissions is providing some useful insights in the way that they are adapting WURF to suite their development approach. Student still submitting incomplete requirements! It should be possible to automate WURF in some way in order to resolve issues regarding its completeness.</td>
</tr>
</tbody>
</table>
## AR3 Observations

<table>
<thead>
<tr>
<th>Date</th>
<th>Location</th>
<th>Observational Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oct 2009</td>
<td>Lab</td>
<td>Major issue with the association system within eWURF. Student indicated that the screen on the association page goes blank after they attempt to define the association. The only way to recover is to close the browser.</td>
</tr>
<tr>
<td>Oct 2009</td>
<td>Lab</td>
<td>Some students are able to add empty objectives to the system, which causes major problem with the automated SRS system.</td>
</tr>
<tr>
<td>Nov 2009</td>
<td>Lecture</td>
<td>Provided a research method lecture to the final year students embarking on their dissertations. Also showed them eWURF. Some web students indicated that they wanted to use previous ‘paper based’ versions (RUMM) (WURF), as they prefer there approach.</td>
</tr>
<tr>
<td>June 2009</td>
<td>Office</td>
<td>Starting to undertake data analysis of student marks vs usage of eWURF (data captured by student usage of online WURF). Preliminary indications show that students used the online tool 7 to 8 times over the duration of the module. I had expected this to be much higher.</td>
</tr>
<tr>
<td>Oct 2010</td>
<td>Office</td>
<td>eWURF registration system still being actively used even though the release phase is now finished. New users traced to Franchised programmes running in Botho College in Botswana and London Tec in Sri Lanka.</td>
</tr>
</tbody>
</table>
# Module Title
On-Line Business Systems

## Module Status
New

## Date of Official Approval
2003

## SITS Module Code
MUL3030-N

## Module Descriptor
This module takes students, who may have no specific computing knowledge, through the theoretical and technical skills required to design and develop dynamic web applications. A practical approach will be adopted throughout and students will be expected to develop a dynamic database driven website utilising appropriate authoring tools and server side scripts.

Students will be assessed individually on their ability to analyse user requirements and develop dynamic web applications.

Students will also be introduced to a methodology to underpin project development. This methodology will be user centred and will include tools to model the user characteristics and build a set of requirements specific to the project.

## Key Words
Web Development, Web Solutions, Dynamic Website

## Module Type
Standard

## Credits (UoT CAMS)
20

## Total Learning Hours
200

## Level
6
<table>
<thead>
<tr>
<th>Summative Assessment</th>
<th>ICA 100%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assessment Marking Scheme</td>
<td>2004 UG Module</td>
</tr>
<tr>
<td>Assignment Submission</td>
<td>School of Computing</td>
</tr>
<tr>
<td>Available as Open/Distance Learning</td>
<td>No</td>
</tr>
<tr>
<td>Assignment Re-submission possible</td>
<td>Yes</td>
</tr>
<tr>
<td>Roll On / Roll Off</td>
<td>No</td>
</tr>
<tr>
<td>Compensatable</td>
<td>Yes</td>
</tr>
</tbody>
</table>

| Pre-requisites | None |
| Co-requisites | None |
| Other Requirements | Forbidden combination with Web Scripting - MUL2021-N. |

**TEESIDE UNIVERSITY QUALITY UNIT APPROVAL OF A MODULE - MODULE SPECIFICATION**

**Module Aims**

This module aims to:

1. Develop an understanding of the programmatic controls for both the content and presentation of web pages e.g. XHTML and CSS-P.
2. Introduce theoretical and practical considerations when working with authoring tools, server-side scripting languages and databases.
3. Encourage the use of development methodologies and analysis tools throughout the development process in order produce a user and client centred solution.
4. Provide students with an opportunity to gain experience in developing web applications using an industry standard integrated development environment.
5. Provide students with the opportunity to design and develop server side scripts and database tables.
6. Highlight the need for continual evaluation, maintenance and marketing during the lifecycle of an online business system.

**Indicative Content**

- Introduction to methodologies for the management of the implementation process. Use of a user modelling method to help understand user requirements.
- Writing XHTML compliant mark-up. Using CSS-P to separate presentation from content. Awareness of the need to address accessibility requirements in the implementation of web based applications.
- Introduction to Server-side technologies, e.g. PHP and MySQL
- Client to server communication model. Data validation and security issues when developing and deploying online business systems.
- Concepts of scripting for the web.
  1. Problem solving and providing efficient solutions using a range of scripting languages, including PHP and Actionscript.
  2. Working with objects, functions, conditionals and loops.
  3. Writing database connection and query scripts.
  4. Data Types, Variables, Assignment, Input and Output.
- Introduction to Flash for the production of multimedia components and applications.

**Learning Strategy**

The module will be taught by a blend of lectures, practical tutorials and online resources. Blackboard will be used to provide learning objects and supporting materials where this is deemed appropriate. Students will be provided with the necessary theoretical concepts in relation to SQL and PHP. Extensive tutorial support is provided for the IDE and to enable the connection to the database for query execution.

Lectures will be used to develop understanding of the underpinning theory and concepts involved in implementing web applications. Active lectures, using a variety of techniques, will enable students to investigate key concepts in relation to the tools within the session itself.

Practical tutorials will concentrate on developing key skills within the implementation environment, including design and application implementation. Students will be expected to dedicate a substantial amount of time working on their web applications outside of class time. This self-directed learning will include research and developing practical skills.

**Learning Outcomes**

*Knowledge & Understanding*

On successful completion of this module, the student will be able to:
1. Explain what is involved in the development and design of a dynamic, database enabled web application using contemporary tools and techniques.

**Cognitive & Intellectual Skills**

On successful completion of this module, the student will be able to:

2. Justify the design and development of web applications and critically evaluate their implementation approach.

**Practical & Professional Skills**

On successful completion of this module, the student will be able to:

3. Design and develop web applications using an industry standard integrated development environment (IDE).

**Key Transferable Skills**

On successful completion of this module, the student will be able to:

4. Document the web application development process.

5. Critically evaluate appropriate methodologies and recognise the need for continual development and marketing of an online business system.

**Assessment**

**Assessment Strategy**

The module will be assessed by one component weighted at 100% that assesses the learning outcomes 1, 2, 3, 4 and 5. Students will receive the ICA documentation at the start of the module and will work towards the ICA throughout the module through the ongoing development of a portfolio. The assessment will include the implementation of a dynamic website and a written report of approximately 2500 words documenting the design and implementation of the website.

Formative feedback is provided throughout the module.

The School operates a standard procedure for providing (at least) a minimum level of feedback to students in line with the University's Assessment and Feedback Policy.

**Assessment Criteria**

Students will submit a dynamic website for a given scenario. The website will assess learning outcome 3.

Students will also be required to submit a report based on the design and implementation of their dynamic website (approx 2500 words). The report will assess learning outcome 1,
2, 4 and 5. They will be assessed on the quality of their design documentation, the justification of their design choices and a discussion of their implementation.

Students will be provided with a detailed assignment brief.

**Indicative Resources**

*Purchase*


*Essential*


*Recommended*


*Journals*

International Journal of Web Engineering and Technology

Interacting with Computers
New Review of Hypermedia and Multimedia

Electronic

University of Teesside, Elearning@Tees. http://eat.tees.ac.uk.

Accessibility

The School aims to make this module accessible to any student who may benefit by studying it. Students who are concerned about their ability to access the module are advised to contact the School Disability Coordinator for academic advice and the University Student Services staff for details of available support.
Appendix A3 – Related Work

3.1 Introduction
Chapter 2 within Volume 1 provided a background to this research programme by outlining some of the problems facing Web, Software and Requirements Engineering. Important theory in respect of RE and SE sets the context for further work in terms of this research programme. This section presents a literature review of related work, subdivided into three sections. It demonstrates how each research cycle reflected the examination of existing approaches in order to solve problems identified from each evaluation.

3.2 Requirement Process, Methods and Tools
A requirements process is underpinned by specific methods and tools that are selected by the development team. Some methods reflect the whole requirements process (elicitation, analysis, specification and validation) and some address one or two stages of the typical RE process. For example, some focus on elicitation or the specification of requirements. It was found that some developers choose to combine methods in order to satisfy particular organisational or problem objectives, thereby creating hybrid methods suited to the organisation. The aim of this section is to demonstrate variations in the approaches that are in use by both academics and practitioners.

It was considered important to undertake a structured analysis of the methods by comparing their treatment against the requirements process as defined by Sommerville & Sawyer 1997 and Berry 2003. Table 3.3 demonstrates how each approach addresses the whole requirements process criteria (elicitation, analysis, specification and validation). A discussion of how the review undertaken in section 3.3 relates to the three action research cycles can be found in section 3.7.
3.3 Review of Existing Requirements Processes, Methods and Tools

There is a strong argument for suggesting that there is no *definitive* or *one-size fits all* requirements process, method or tool that can be adopted. “Of course, there is no “one-size-fits-all” approach to this critical front-end challenge and development teams will likely is a combination of techniques for any particular circumstance” (Leffingwell and Widrig, 2010). Much depends on the context in which the software is being produced, the organisational expertise in Requirements Engineering and constraints experienced by the organisation such as human resources and budget. It also depends on the approach taken in terms of the development methodology, where agile approaches need to be supported by agile requirements. The aim of this research is to investigate how to bridge the gap between RE and WE by proposing a web user requirements approach aimed to support the inexperienced student user. As such, it is deemed advantageous to consider what already exists in this domain in order to address specific contextual needs. Although there are no organisational constraints there are time pressures from the perspective of the students’ support and learning needs, which will influence the proposal of a web user requirements approach.

A number of specific methods and tools can be found in the literature that assists the developer with their requirements discovery, elicitation, analysis and specification. Some of these focus on one stage in the process, whilst others cover the whole process. It should be emphasised that there does not seem to be a definitive method dealing with the *elicitition, analysis* and *specification* of web requirements in an educational context. Much of the research therefore investigated methods within the Software Engineering (SE) domain. Whilst it is recognised that there are differences between WE and SE, there are key principles that can be applied to both, thereby contributing to understanding of how RE can be applied to WE. This section investigates and
analyses existing methods within SE and a limited number of WE specific approaches.

<table>
<thead>
<tr>
<th>Method</th>
<th>Requirement Process Stages</th>
<th>Elicitation</th>
<th>Analysis</th>
<th>Specification</th>
<th>Validation</th>
<th>Meets All Criteria (elicitation, analysis, specification and validation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM</td>
<td></td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>▲</td>
</tr>
<tr>
<td>OVID</td>
<td></td>
<td></td>
<td>■</td>
<td>■</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCA</td>
<td></td>
<td>■</td>
<td>■</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TBAS</td>
<td></td>
<td>■</td>
<td>■</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CI</td>
<td></td>
<td></td>
<td>■</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>US</td>
<td></td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>▲</td>
</tr>
<tr>
<td>MSF</td>
<td></td>
<td>■</td>
<td>■</td>
<td></td>
<td>■</td>
<td>▲</td>
</tr>
<tr>
<td>NDT</td>
<td></td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>▲</td>
</tr>
<tr>
<td>JAD</td>
<td></td>
<td>■</td>
<td>■</td>
<td></td>
<td>■</td>
<td></td>
</tr>
<tr>
<td>CRC</td>
<td></td>
<td>■</td>
<td>■</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC</td>
<td></td>
<td></td>
<td>■</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AWARE</td>
<td></td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>▲</td>
</tr>
<tr>
<td>URN</td>
<td></td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>▲</td>
</tr>
<tr>
<td>SOARE</td>
<td></td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>▲</td>
</tr>
<tr>
<td>SSM/ICDT</td>
<td></td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>■</td>
<td>▲</td>
</tr>
</tbody>
</table>

Table 3.3 Comparison of Requirements Methods.
3.4 Related Work Research Cycle 1 (RUMM)

3.4.1 AWARE Approach

Work undertaken by Bolchini and Paolini, 2002 and 2004 into ‘Goal-Driven Requirements analysis’ provides some useful direction for a student focused web user requirements approach. Bolchini’s work culminated in the development a model for the Analysis of Web Application Requirements (AWARE). This is described by Al-Salem 2007 as “a light-weight methodology based on the ‘i*frame-work’, which blends goal-directed RE and scenario-based techniques for defining hypermedia requirements concerning aspects such as content, interaction, navigation and presentation for Web applications” (Al-Salem and Samaha, 2007). An interesting point to note here is the reference to the ‘i* framework’, which can be described as a user modelling approach. It was first proposed by Eric Yu in his doctoral thesis and later refined in his research into “Early-Phase Requirements Engineering” (Yu, 1997). This work proposes that ‘agents’ which are composed of multiple users can be modelled in terms of their characteristics and attributes such as goals, tasks and abilities. The ‘i* framework’ is now part of an international standard for user requirements notation (URN).

AWARE is user orientated, concentrating on high level communication goals in relation to the design aspects of the web application. The AWARE meta-model, (see Figure 3.4.1), contains constructs that enable developers to build up a rich picture of the web application to be developed. A construct model helps define these in more detail, for example, a ‘Stakeholder’ construct defines a set of users, expressed as Actors, with these described in further granularity of detail via the user profile. Another construct named Goal provides a mechanism to express informal stakeholder goals. The Goal construct additionally provides a way of classifying relevance in relationship to other goals, using a hierarchy of numerical or alpha numeric notations. An
actor may also have a varying priority, reflecting the actors that have more importance within the web application. The meta-model enables relationships between the goals and actors to be defined.

The Task Construct builds and refines the actor goals into high level user activities. A Requirement Construct is composed of refined goals and tasks in a natural language that can be understood by all those involved in the web application design and implementation. The meta-model does not imply how to capture all requirements with only those considered critical in shaping the user experience listed in the documentation.

Further, the meta-model links requirements to conceptual design specifications using what could be described as agile techniques. “In particular, the following issues are addressed: a) high-level communication goals and user requirements have to be taken carefully into account in the requirements analysis; b) requirements should be tied up coherently with conceptual design of hypermedia specifications; c) Requirements Engineering techniques should be extremely lightweight, intuitive and usable by web analysts; d) the model proposed must show a relative advantage to project managers, requiring little training effort to be adopted and effectively integrated into current practices” (Bolchini, et al., 2003).
Work undertaken by Bolchini and Eric 2004 provides an alternative approach with the identification of the user profile as the starting point of the user requirements analysis. “A user profile describes stable archetypal qualities of a relevant target user segment (Carroll, 2002) and may comprise a variety of attributes based on demographic, for example, age, gender, occupation, disabilities etc. or “webographic” eg. net usage habits, interests and software constraints, favourite sites etc” (Garrett, 2002). Profiles can be discovered through a variety of requirements elicitation techniques based on user research, such as surveys, contextual inquiry, focus groups and structured interviews. “The user profile can be broken down into four elements; a person profile, a web use profile, a context of use profile and personalising the user.
The profile should be as accurate, clear and realistic as possible” (Bolchini et al., 2003).

### 3.4.2 Usability Context Analysis

UCA is described by Macleod 1994, as an informal approach, due to the fact that the process is not represented by any recognised notation or universally accepted diagrams. “Usability Context Analysis (UCA), is a practical cooperative method for identifying and recording contextual aspects of usability in system development or redesign and for helping ensure that usability evaluations reflect the context of use” (Macleod, 1994). UCA was born out of a project run by UK National Physical Laboratory (NPL) and HUSAT Research Institute as part of the MUSiC (Measuring Usability of Systems in Context) toolkit. In the context of research into a suitable approach for web user requirements one of the interesting aspects of this approach is that usability and evaluation are explicit within the process. A mechanism to evaluate what is produced at the end of the project is beneficial as it demonstrates the traceability of requirements to the web design. “There are two prerequisites for any evaluation, if it is to have valid and useful results: the data must be drawn from circumstances which have acceptable ecological validity; and an appropriate method must be applied for analysing the data” (Macleod, 1994). By eliciting, analysing and documenting requirements, students would have a valid set of requirements in which to base their evaluation. The method in which this could be achieved requires further investigation in the research programme.

Usability Context Analysis does involve a high number of steps and the involvement of a number of people in order to successfully complete it. One of the steps within the method is the ‘Context Questionnaire and Guidance’. The questionnaire attempts to capture ‘User Types’ and classifies users as Primary and Secondary. The modelling process is interesting as it allows the
production of a profile of the user using the headings: *Skills and Knowledge, Physical Attributes, Mental Attributes, Job Characteristics and Tasks*. The method uses natural language to build the profile which then flows into the evaluation plan in the form of measurable criteria. The evaluation headings include Users, Tasks and Environment (organizational, technical and physical). It is felt that this aspect of UCA warrants further investigation.

UCA provides a well formed structured approach to the requirements elicitation and analysis process. It pays particular attention to the users, classifying them as ‘Types’ which are then modelled as generic entities for the evaluation process. UCA could be partly defined as a cognitive aid, as it helps in the logical process of linking assumptions based on documented facts, forcing consistency and traceability of the requirements through to the web design. Perhaps the most interesting ideas evident in this method are ‘user types’ and ‘profiling’. This would be something that students may readily identify with, especially as it uses natural language to describe user characteristics.

The requirements classification system used by UCA is based upon functional and non-functional requirements, along with a meta-model to support design requirements, usability requirements, user interface requirements and features of the system that the user interacts with.

### 3.4.3 Contextual Enquiry

Bolchini’s approach is to develop an in-depth understanding of the target users. One of his methods is *contextual inquiry (CE)*. “Contextual Enquiry is a technique for examining and understanding users and their workplace, tasks, issues and preferences. CE can be used to produce user needs analyses and task analyses. The results of a CE feed directly into the design process” (Gaffney, 2007).
A web user requirements approach aimed at students could adapt CE to form part of a learning exercise, where the tutor acts as the target user and the student asks a number of questions to elicit the required information. The information could then be captured within a user profile. “Profiles can be discovered through a variety of requirements elicitation techniques based on user research, such as surveys, contextual inquiry, focus groups and structured interviews. The user profile can be broken down into four elements: a person profile, a web use profile, a context of use profile and personalising the user. The profile should be as accurate, clear and realistic as possible” (Bolchini and Eric, 2004).

Bolchini highlighted profiling as a vehicle for the way that user requirements are presented. A person profile is perhaps the most interesting aspect from the point of view this research and as way forward for a student centred learning activity. A profile could include the characteristics of the user, including age, gender, motivations, preferences and skills. Having formed a mental model of the user, the student could then think much more deeply about user requirements. In common with the two previous approaches, contextual inquiry does not cover the entire requirements process. Most of the activity concentrates on the elicitation stage. In addition, little attention is given to analysis and it is not clear how functional and non-functional requirements are treated.

### 3.4.4 Usage Scenarios

This method relies on user involvement in the requirements process by eliciting information from them directly. “Usage scenarios and the use cases from which they are derived should describe an application's expected behaviour in multiple business processes. Gathering this information requires sitting down with users and determining every possible scenario for a specific function” (Cornish, et al., 2003). This method works best where the
complexities of the application arise from multiple processes which need to be constructed. By writing a usage scenario with the user, the development team can focus much more on the problem.

The scenarios can include steps, events and actions which occur during the business process. This makes the method quite flexible in terms of interfacing with more complex system modelling techniques such as UML or other system development methodologies. “Usage scenarios are applied in several development processes, often in different ways. In derivatives of the Unified Process (UP), such as the Rational Unified Process (RUP), ICONIX and the Agile Unified Process (AUP) they are used to help move from use cases to sequence diagrams. The basic strategy is to identify a path though a use case, or through a portion of a use case and then write the scenario as an instance of that path” (Ambler, 2004). The Usage Scenario process involves writing a conceptual description of what the application must do. This is then translated into the conceptual description in a logical representation of the design, comprising of components in the application, often carried out using ‘use cases’.

Usage Scenarios help with an often neglected stage within the requirements process, *elicitation*. It achieves this by gathering information directly from the user and represents this in the design using system modelling techniques. In an educational context, the fact that usage scenarios rely on direct involvement with a user may detract from its usefulness, although again the tutor could *act* as a user for this purpose either in class or by providing information in the assessment documentation. In terms of requirements classification, Usage Scenarios concentrate on functional, rather than non-functional requirements.
3.4.5 Key Ideas That Emerged From The Review (Research Cycle 1)

Having undertaken a review of related work in the first cycle, a number of ideas formed concerning an initial web requirements method aimed to support inexperienced student:

• Students need to define different types of requirements, such as: Navigation; Content; Colour; Structure/Layout; that represents the design aspects of a web design.

• Encourage the Student to Model the User through profiling techniques as the starting point in the requirements discovery process. The user could be defined as a primary or secondary user, that encourages the student to think about the roles that the need to be considered.

• A lightweight method that supports students design decisions. Encourage the student user to make connections between requirements and the design, thereby making the traceability of requirements explicit and assessable in the ICA submissions.

• An enquiry based method may enable the student to envision a rich picture of the project.

• Requirements discovery encourages the student to analyse the problem domain more thorough and to document these in design centred requirements.

The ideas were more thoroughly developed in the first research cycle, see Volume 1, chapter 6.
3.5 Related Work Cycle 2 (WURF)

3.5.1 Object View and Interaction Design (OVID)

Object View and Interaction Design (OVID) (Roberts, et al., 1997) cannot be described as a ‘whole process requirements method’, as it concentrates upon matching tasks to interface objects. It represents this through class diagrams, but it is not clear how elicitation takes place. OVID is an example of a user centred approach, as its main aim is to bridge the gap between the implementers and end users. “In interface design, we call the models in the users heads the ‘user's model’. In our designs, we try to take advantage of how users employ this model. There are also two other models involved in product design, the designer's model and the implementation model. The designer's model is what users are supposed to see when they use the product and the implementation model is what the implementers actually program” (Roberts, et al., 1997).

OVID provides a way forward to link tasks to user driven interface objects whilst recognising that there are differences between design and development. However, it is argued that OVID focuses too much on this aspect. “OVID is a method for representing requirements during the analysis and communication phases. It assumes that the acquisition phase has already taken place and indeed some analysis in the form of tasks. The output of the method is an abstract diagram that describes the architecture of the desired design, from the users' point of view. The diagram is used in conjunction with the visual specifications to enable implementation of the final diagram” (Berry, et al., 2007).

OVID is focused on the User Interface, something that use case modelling does not specifically address. As a result, the design may progress more quickly with fewer cycles of iteration. This is something that could be tested
in the design and evaluation of an approach aimed to support the inexperienced student user. “OVID specifies the interface design in a format and notation well-suited for code design and feeds directly into tools and methods commonly used for programming, thus reducing the risk of introducing interface errors later in development” (Berry, et al., 2007).

OVID uses natural language to describe interactions, tasks and objects, but still draws upon technical approaches in the form of class diagrams. These diagrams can be described as abstract or conceptual as their interpretation is not yet represented in the visual design or behaviours linked to specific interactions. The requirement classification system employed by OVID is based around the term functional and non-functional requirement.

3.5.2 Agile Requirements Method
The Agile Requirements Method (ARM) (Leffingwell and Widrig, 2003) brings together a range of tools in a flexible way. It starts by envisioning a concept for the overall project. A concept is the root of the project, which might be fully understood at the outset, but might be an idea that needs to be explored more fully. This is achieved through workshops or interviews undertaken with end users and other stakeholders. A ‘vision’ for the project is a document that describes the features to be implemented. Requirements are then refined using a use case diagram, with each use case containing its own specification template. The classification system used by the Agile Requirements Method utilises the functional and non-functional definitions. ARM also recognises the need for a management of requirements, but leaves this open in terms of the tools which would be used to support it. Project management is also a feature of this method, reflecting the need to mesh with other processes that the development team might be using.
Figure 3.5.2 Agile Requirements Method (Leffingwell and Widrig, 2003).

According to Leffingwell and Widrig, the Agile Requirements Method suits projects that are ground breaking in terms of innovation and the tools that are used to control requirements. It could be described as a complex approach, as some stages are explicit in terms of the tooling that can be used to achieve it, but other stages are left open. For example, the vision stage offers no template could be used by the developer. This may elicit advantages and disadvantages if it was adopted in an educational context. Students could pick and choose from a selection of tools that best fit particular projects, but with the underpinning of a process that is documented and repeatable. Some students might find this openness difficult and choose to complete the stages superficially or not at all. Nevertheless, ARM provides some useful ideas that could contribute to the development of a student focused web user requirements approach. In particular, the vision document is used as a starting point and could be useful in the initial stages of an approach, as it enables students to see a richer picture of the web project.
3.5.3 Task Based Audience Segmentation

Thus far this section has focused on three academic approaches. A commercial requirements approach could provide a useful contrast to the approaches discussed earlier. Task Based Audience Segmentation (TBAS) is a commercial approach that helps the developer to think about the user and the tasks they complete in order to achieve a goal. “It is an invaluable foundation for conceptual research, that is to say it can reveal how people think about completing online tasks” (Young, 2007). TBAS cannot be defined as a true requirements method as it does not include an elicitation stage. Instead it concentrates upon the analysis stage, where it reveals users and their needs by the tasks that they undertake. In terms of requirements classification, both Functional and Non-Functional requirements are absent from this particular method.

In order to carry out TBAS, a number of steps are required as described by Young, 2007:

1. Team Preparation.

2. Brainstorming (tasks, goals and users).

3. Task Grouping (to create user goal sets).

4. Link Users (Audience) to Tasks.

5. Define Audience Segments.

The TBAS method encourages the developer to think about tasks and users and to link a user to specific a task. After an initial brain storming session, groupings are used for tasks which are then matched with users to create a conceptual map. Relationships between users and tasks are then identified and named accordingly, using mainly marketing terminology. This activity may prove useful in an educational setting, where students could undertake
brainstorming activities to think about the end users and their tasks. Linking a task to a user would also enhance the traceability of requirements within the overall process.

3.5.4 Agile Microsoft Solutions Framework (MSF) – Requirements Stage

According to Anderson 2005, MSF enables the software development team to follow a proven Software Engineering lifecycle. It has an adaptable meta-model, which contains rules, frames and constraints, that can be adapted to suit the type of application being developed. The need for a flexible approach is due to the divergences in software development, such as agile, rapid and prototyping. “Many software developers are suspicious of process generally. Process often gets in their way and slows the pace of software development to a frustrating level” (Anderson, 2005). This idea is reinforced by Leffingwell and Widrig, 2010, who cite the reason for developers wanting more flexible approaches as “failures in the waterfall model, along with increasing time-to-market pressures and advances in software development tools and technologies” (Leffingwell and Widrig, 2010).

MSP achieves a flexible or agile approach by providing specific process guidance and templates which map onto the adaptive process. These stages include; Envisioning; Planning; Developing; Stabilising and Deploying. One of the stages within MSF that relates to this research investigation is planning. Within the planning stage, a number of templates enable the production of a detailed overall specification for the project. These help formulate highly detailed requirements and include “Business Requirements; Conceptual Design; Functional Specification; Logical Design; Operations Requirements; Physical Design; System Requirements; Usage Scenarios; and User Requirements” (Anderson, 2005).
What is interesting from the planning stage is the separation of business requirements from user requirements. The planning stage within MSF broadly reflects the requirements process, although excludes requirements management. Usage scenarios are separated from user requirements. In many of the previous approaches, requirements were grouped together as one set including; *functional requirements, tasks, goals, usage scenarios and UI behaviours*. MSF classifies requirements into; *Business Requirements, Operations Requirements, System Requirements and User Requirements*.

The flexible meta-model approach embedded within MSF allows for adoption or rejection in any of these specification or requirements templates, essentially allowing the developer to *pick and mix* meta-models to fit the project. In terms of the design of a web user requirements approach aimed at students, this approach has potential. For example, it could allow the student to adopt an appropriate meta-model to fit their project. Where a set of requirements are not applicable, for example, a set of non-functional requirements, they would not need to appear in the final specification.

### 3.5.5 Use Case Diagrams and Templates

Use Cases are part of UML, which is a notation for modelling software systems. “UML expresses system models and designs in an object-oriented fashion” (Conallen, 2003). Use Case Diagrams are used as starting point in the analysis of requirements within a number of contemporary requirements methods. For example, webRE, from which requirements are expressed in more formal notations including NDT patterns and UWE activity diagrams. “Use cases help to determine the functionality and features of the software from the users perspective” (Escalona and Aragon, 2008).

A Use Case helps to describe the goal of the user by the definition of steps required to achieve it and the use case diagram provides a graphical
relationship of all the Use Cases. The graphical aspect is important as it allows the analyst to *visually* depict the main functions to be implemented within the software. Users are represented by the term ‘Actor’ and are typically represented in the use case diagram by a ‘stick figure’. Relationships between the Actor and Use Case are represented by a drawn line, with multiple actors and multiple relationships often illustrated.

Actors can be further defined as having a *primary* or *supporting (secondary)* role within the system. “The primary actor of a use case is the stakeholder that calls on the system to deliver one of its services” (Cockburn, 2001: p54). Supporting actors are usually external to the system that is being developed, for example, a web service or hardware such as a printer. The supporting actor is important in a web project, as it possible to interact with multiple external systems. Figure 3.5.5 shows a typical Use Case diagram with associations between actors.

![Figure 3.5.5 Example Use Case Diagram.](image)

300
A *Use Case Template* provides a mechanism to further refine the use case including the flow of events triggered by the function and its pre-conditions. A use case may also be useful for early development or brainstorming of initial system requirements. “It can be used to focus discussion about upcoming software system requirements, but not to be the requirements description” (Cockburn, 2001: p7).

Within a web user requirements approach, Use Cases could provide a way of defining users who will interact with the website. In particular, the way that Use Cases classify users *(primary and secondary)* would allow students to model their users and to express their profiles in a universally recognised language.

**3.5.6 Navigational Development Techniques (NDT)**

NDT has been developed specifically for use in web development in order to capture, analyse and specify web requirements. NDT is the outcome of extensive research at the University of Seville, by Maria Escalona and Gustavo Aragon in 2008. NDT is a method that reflects all stages within the requirements process. It achieves this by offering a strict workflow using ‘Model Driven Web Engineering’ (MDWE). “MDWE proposes representing concepts using meta-models. The development process is supported by a set of transformations and relations between concepts that leads to agile developments and assures consistency between models” (Escalona and Aragon, 2008). Using this workflow the requirements are acquired and then defined according to their nature. This includes a three stage process, with some guidelines and heuristics to help during their production. These stages include:
Stage 1 Requirements Capture (*)

1. Information Storage Requirements.
2. Actors Requirements.
3. Functional Requirements.
4. Interaction Requirements.

Stage 2 Requirements Definition

1. Content Model (using a class diagram).
2. Navigational Model (represented in a navigation chart).
3. Abstract Interface Model (showing part of the web user interface using prototypes).

Stage 3 Requirements Validation

1. Evaluation of Prototypes.
2. Requirements Specification to be used in Web Methodologies to deal with Design and Implementation.

Figure 3.5.6

Navigation Development Techniques Model (Escalona and Aragon, 2008).
It is felt that this method has enormous potential for Web Engineering and for teaching Web Engineering due to the transparency of the stages and its ability to link with a number of existing web methodologies. The method does not add unnecessary complexity and tools used within the method include Use Cases; Navigation Mapping; and Entity Relationship Diagrams. In common with other approaches, the user is expressed as an actor within the web application. The framework deals with this within an NDT package named ‘Behaviour’. The WebActor Class models in detail their behaviours, attributes and their relationships. Functional requirements are represented by a WebUseCase class. Navigational activities are represented by Browse, Phrase and Transaction classes.

In terms of a web user requirements approach, the teaching of classes and class diagrams would suit many web development modules that deal with databases, as their fundamental concept is an embedded feature of the curricula. That does leave a gap, where students studying web design modules may find it difficult to engage with the notion of classes and diagrams to represent the requirements model. This cohort of students represents a significant overall number on web modules within the HEI.

3.5.7 Joint Application Development (JAD)

Joint Application Design (JAD) was written and developed by Drake, Josh and Crawford of IBM. The principle idea is to bring together developers and users together in a productive and creative setting. JAD overlaps RE and SE, where both requirements and analysis are undertaken. Nevertheless it is still felt that there are some stages within JAD that could inform the structure of a web user requirements method. “JAD is a method whereby system stakeholders work together in facilitated group sessions to specify and perform preliminary development (Requirements Engineering and analysis) of a system. JAD sessions include representatives in the following roles:
session leader (facilitators), user representative, specialist, analyst, information systems representative, executive sponsor” (August 1991).

A JAD session starts with the *business vision* and moves onto the definition of *high level requirements* and the setting of *business objectives*. It attempts to capture both functional and non-function requirements in the same session, including security requirements and constraints. It also supports the documenting of requirements in addition to analysis and modelling. JAD relies heavily on facilitation and stakeholder involvement and when used on projects can take days to complete. “Knowledge workers and IT specialists meet, sometimes for several days, to define and review the business requirements for the system” (Haag, *et al.*, 2006).

Setting an overall vision for the project, along with measurable business objectives is an interesting aspect of JAD and again this is something that could prove beneficial to a web user requirements method. Students would be able to see a richer picture of the project and would be able to think about what the website hopes to achieve by expressing its business objectives.

### 3.5.8 Cooperative Requirements Capture (CRC)

The principle concept behind CRC is that mathematical and technological notations are insufficient in reflecting user requirements. According to Macaulay, the social element is “explicitly managed through use of a human facilitator and which provides a structured approach to the management of the requirements capture task. The process comprises seven stages; 1. The business case; 2. Workgroups; 3. Users; 4. Objects; 5. Tasks; 6. Interactions; 7. Consolidation” (Macaulay, 1993).

The process encourages the production of documentation in each stage although it does not provide a template for this. The process uses the idea of *user tasks* to help the stakeholders think about requirements. A hierarchy of
tasks are produced and their interactions associated to specific users. The consolidation stage allows stakeholders to revisit the business case and reassess the credibility of the information gathered.

However, it is not clear what happens after these stages have been completed. For example, it offers little advice on the analysis and specification stages. CRC is therefore focused on the elicitation stage. CRC could be useful in helping define the elicitation stage for web user requirements. For example, the way that CRC focuses on ‘cooperative interaction’ between stakeholders in drawing out requirements may prove useful. Obviously this would need to be simulated by the tutor, rather than interacting with the users themselves.

3.5.9 User Requirements Notation (URN)

URN is supported and ratified as an international standard by the International Telecommunications Union, a United Nations agency for information and communication technology issues. This is a body that has responsibility for co-ordinating the global use of the radio spectrum and for assigning satellite orbits and so is very well recognised in various fields. The fact that URN is now a ITU standard underlines the importance of work into requirements and the way in which it should be approached.

URN is able to model functional and non-functional requirements by adopting the Goal-Requirements Language (GRL) to model tasks and procedures. It achieves this by the use of scenarios to draw out ‘high level’ functional and non-functional requirements. “The URN is a two-headed proposal. URN-NFR addresses non-functional requirements (NFRs), capturing them using the Goal-Requirements Language (GRL). Such a model aims at highlighting how some facets of a system (e.g., tasks, procedures) contribute (positively or negatively) to the satisfaction of NFRs” (Arnold, et al., 2010).
As discussed in section 2.2.1, high level requirements are generalised descriptions that are written in a natural language and can be understood by all stakeholders involved in the project. In URN these are then transformed and organised into textual ‘use cases’. “URN combines modelling concepts and notations for goals (mainly for non-functional requirements and quality attributes) and scenarios (mainly for operational requirements, functional requirements and performance and architectural reasoning). The goal sub-notation is called Goal-oriented Requirements Language (GRL) and the scenario sub notation is called Use Case Map (UCM)” (ITU-T, 2008).

Within URN, a ‘topmost’ meta-model describes the attributes, relationships and constraints between the models (see Figure 3.5.9).

![Figure 3.5.9 URN ‘Topmost’ Meta-model.](image-url)
In addition, the GRL meta-class allows the developer to model the goals and intentions of the stakeholders, with the latter modelled as actors. URN would be difficult for students to adopt and use for a number of reasons. URN is quite a complex modelling approach, using a language that design students would not be familiar with. However, this would bring an additional overhead in terms of learning for students before they could use it. It would also require an overhead in terms of time and resources in order to support it. It cannot be therefore described as a ‘light weight’ approach that matches the agile requirements method that is needed to support the inexperienced student user. Whilst this is true, there are some useful aspects within URN that could be explored further. This includes the ability to profile the characteristics of the users (actors); the way that it is able to model tasks and the association of tasks to actors, represented within the meta-model.

3.5.10 SOARE Approach

Bleistein, et al., 2004 have also investigated web requirements and produced an approach named ‘Strategy-Oriented Alignment in Requirements Engineering’ (SOARE). This provides a basis for defining ‘e-business web requirements’. SOARE is aimed at e-business projects that are both web and non-web based. It focuses on business objectives and the ‘real world goals of the system’. SOARE starts with the identification of business objectives as a means of decomposing requirements. “The SOARE approach incorporates means for analysing and decomposing business strategy, employing goal modelling both to represent business strategy in a Requirements Engineering context and to link high-level strategic objectives to low-level requirements through goal refinement” (Bleistein, et al., 2004). It also provides traceable links between business objectives and goals, making these measurable within the end product. Business strategies are expressed in a natural language and derived from business plans, annual reports, stakeholder interviews and
executive reports. This approach is very different from AWARE, as it is business focused and uses language that requires some background understanding of business strategy and planning. SOARE offers a model for the requirements process, as illustrated in Figure 3.5.10.

![Figure 3.5.10 SOARE Process (Bleistein, et al., 2004).](image)

**3.5.11 SSM/ICDT Approach**

Meldrum and Rose, 2004, argue that “there is a need for an approach to requirements generation for web systems that combines the recognition of multiple user views of a complex human activity system with techniques to help creatively map existing and potential business functions to a Web-based environment” (Meldrum and Rose, 2004). This view supports the idea that an approach must be accessible to those that have no IT background and who do not have an understanding of formal notations to express requirements, or other I.T specific terminology. It employs Soft Systems Methodology (SSM) to deliver various meta-models that demonstrates how describe business processes relate to human activity within the web application. Ramesh, et al., 2002 in Meldrum and Rose, 2004, identifies the difference in characteristics between software and web development. “These can include time pressure, vague requirements, a prototyping orientation, frequent releases and evolutionary development, parallel development and an emphasis on small teams of highly competent programmers coding their way out of problems” (Meldrum and Rose, 2004).
The process model, see Figure 3.5.11, starts with the analysis and expression of a problem faced by the developer. “Using rich pictures and root definitions and conceptual models of relevant human activity Systems” (Meldrum and Rose, 2004).

Figure 3.5.11 Web-site development using SSM, the ICDT model and prototyping (Meldrum and Rose, 2004).

The activities are mapped onto a requirements analysis matrix and categorised into information, communication, transaction and distribution (see Figure 3.5.12). “Each activity is then analysed using the question ‘How could a Website support the information/communication /distribution /transaction potential for the activity? The answers can be mapped onto the matrix resulting in a creative set of potential ideas. These can then be prioritised (in terms of feasibility and desirability) in discussion with users to arrive at an agreed set of requirements. This is followed by a prototyping development strategy” (Meldrum and Rose, 2004: p195).
3.5.12 Key Ideas That Emerged From The Review (Research Cycle 2)

The second review was undertaken in the early stages of research cycle 2 in order to address weaknesses identified in the evaluation of the experimental method. The context for the review was informed by the evaluation of the first research cycle. A number of ideas emerged that informed the re-working of the method into a more fully developed framework:

- More emphasis on the ‘discovery phase’ by focusing more on the vision behind the website, its objectives and the tasks that the user will perform. This replaced the ‘user profiling’ aspect of the first method, which proved to be confusing for the student.

- In order to address the main weakness of the experimental method, which was the ‘closed taxonomy for requirements’, it was thought that it would be more appropriate to break requirements into functional and non-functional. Functional requirements would reflect the web project in terms of tasks and features to be built, in which the student would better relate to. Dynamic ‘server-side’ requirements could also be considered in terms of functional requirements, as it is possible to describe their
interactions and behaviours. Design type requirements could still be represented in non-functional requirements, where these impose constrains on the web design and implementation environment.

- A number of key terms cited in the method were changed in order to help the student. The term ‘user’ was dropped in favour of ‘actor’ in order to better represent both human and non-human agents who interact with the website. A change to the meta-model in this regard makes it possible to model more than two actors, again something that was identified as a weakness in the first method.

- The addition of an association model. To make explicit that there is a link between actors, tasks and functional requirements. A direct relationship between the requirements and traceability to the website could then be achievable.

3.6 Related Work Cycle 3 (WURF / CAWE)

3.6.1 Computer Aided Web Engineering

The first method was deployed to the student in a paper based format, where this was completed and submitted along with the students ICA. In the second cycle of research, it was deemed advantageous to use an electronic means of deploying the document, whilst also supporting the student with examples in use. These were packaged for distribution within the institutions virtual learning environment (VLE). In the final research cycle the problem of deployment was once again considered with a premise of integrating more support mechanisms. One of the options for deployment was to re-engineering WURF into a computer based program in which the students could access continually throughout their web project. This approach is a well established in various fields of computer science, where they are referred to as Computer Aided Software Engineering (CASE) tools.
CASE tools can support a variety of stages within the software development lifecycle. A similar acronym can be found in the literature for Web Engineering, although the frequency of citation is much lower than that of CASE. Computer Aided Web Engineering (CAWE) is used to describe tools that help support one or more stages within the web development lifecycle, such as translating designs into code.

The approach offers solutions raised in the evaluation of the second research cycle, such as the consistency, completeness and correctness of requirements that the students are producing. It was found that students were able to produce incomplete requirements documentation in their ICA’s. By adopting a CAWE tool approach, a strict rules model could be enforced thereby ensuring students submit complete requirements. It also offers a way to automate the associations between actors, tasks and functional requirements in order to enhance the traceability of requirements through to the end website.

### 3.6.2 WebRatio

Casteleyn, *et al.*, 2009, describe a number of CAWE based tools such as WebRatio and VisualWade that support model driven web development. WebRatio provides support for five main web development areas *Data Design, Hypertext Design, Data Mapping, Presentation Design and Code Generation* (Casteleyn, *et al.*, 2009, p232) (see Figure 3.6.2).
Figure 3.6.2 The WebRatio Development Process (Casteleyn, et al., 2009: p233).

The goal of WebRatio is to automate code generation by modelling the data, logic and presentational design phases. It is also able to produce ‘java server pages’ (JSP) templates that can be refined further in order to modify presentational designs or data interactions. WebRatio supports rapid prototyping, thus shortening the overall lifecycle, which is useful in web projects due to the short time to market expectations. Unlike other prototyping tools, where mock web pages have to be recoded, WebRatio generates code that can be used in the production version of the website.
3.6.3 FlashWeb

FlashWeb, (see Figure 3.6.3), is a CAWE tool that focuses on data management as a mechanism to model a web application. “The FlashWeb CAWE tool supports the model-driven development of web applications that provide advanced data management functionality. It utilises graphical models throughout the entire development process. Different aspects of the web application are captured with different models assuring a clear separation of concerns” (Jakob, et al., 2007).

![Figure 3.6.3 FlashWeb Development Process (Jakob, et al., 2007).](image)

FlashWeb provides a good example of how a CAWE tool can provide support to web developers by helping them to model various aspects of the web application, before implementation commences. It shares some similarities with WebRatio, such as the ability to model aspects of the web application including a graphical representation, which then informs the code generation process. FlashWeb aims to provide a rapid development environment, but still focuses on the modelling of data, presentation and interactions. However, it does not consider requirements first, although the process model of FlashWeb does acknowledge requirements analysis.
WebRatio, FlashWeb and other related tools tend to focus on implementation, rather than the early phases of the web development lifecycle. “There are many tools available to support the building of web sites, but few that support their planning or design” (Griffiths, et al., 2004).

### 3.6.4 VOLERE Requirements Template

An opportunity to use the CAWE tool to enhance the SRS exists via production of an automated SRS document. This could be achieved by using the stored user data to populate an SRS template, after consistencies and completeness checking had been completed by the rules model. Requirements could also be attributed with a unique identification number for tracking purposes. A summary of the Statement of Purpose, Web Objectives, Tasks and Web Actors could also be included within the SRS. By standardising the specification, it is also envisaged that the traceability between the requirements and website implementation would become more apparent during the marking and feedback process, and for validation purposes.

It was found that a number of requirements specification templates existed, including a number of commercial offerings. For example, VOLERE (Robertson and Robertson, 2010) is a commercial specification template that comprises the following elements:

1. Project Drivers.
2. Project Constraints.
3. Functional Requirements (FR).
4. Non Functional Requirements (NFR).
5. Project Issues.

(Robertson and Robertson, 2010)

The interesting point about the requirement template is that it offers a standard way of setting out Functional Requirements (see Figure 6.2.4). The
designers refer to this as the ‘requirement shell’, which acts as a template for requirements.

VOLERE was not found to be an automated tool, but offered a way forward for a requirements template to be expressed within WURF. In particular, as the VOLERE template allowed for the standardisation of information, this would have benefits not only for the student user, but also the facilitator, in their marking and feedback process.

### Figure 3.6.4 VOLERE SRS Requirement Template (Robertson and Robertson, 2010).

VOLERE was not found to be an automated tool, but offered a way forward for a requirements template to be expressed within WURF. In particular, as the VOLERE template allowed for the standardisation of information, this would have benefits not only for the student user, but also the facilitator, in their marking and feedback process.

#### 3.6.5 SRS Requirements Template

Another example of an SRS document discovered during this research was one in use by a major United Kingdom public sector organisation (see
Appendix B1.1). This document did not have a classification for a functional requirement and merely lists requirements in alphabetical order, using written natural language statements, along with references to additional resources. The SRS did include a version control mechanism using date tracking. This would suit a situation where the requirements are being constantly updated in a team project, but is nonetheless a valuable feature of the SRS. It also includes a mission statement and a project description as an executive summary.

In the two SRS examples reviewed, it would be seen that differences in approach occur, perhaps relating to specific organisational needs. Both reference requirements, either as features, or by a singular reference ‘requirements’. Templates can consist of simple text and bulleted lists or more complex layouts that contain the information. Some specifications also included additional information and links to resources such as a screen shot of a high fidelity prototype.

Although the term ‘SRS’ is software centric, it was expected that some of the examples addressed in this section could be adapted for use within WURF. In particular, an executive summary that includes the Statement of Purpose, Web Objectives, Tasks and Web Actors, would have provided a useful overview of the project to a third party. A more formal template for functional and non-functional requirements would then follow, each with a unique identifier to facilitate tracking and traceability.

Consistency, completeness and correctness are all established and well understood topics within RE and are often cited as the ‘three Cs’ of requirements in the literature. “Consistency requires that no two or more requirements in a specification contradict each other” (Zowghi and Gervasi, 2003). Correctness relates to the needs of the user, their tasks or business
objectives and the corresponding requirement in much the same way as being able to trace requirements to the physical design of a website. Completeness reflects the need to include all requirements within the SRS and the necessity that all the information contained within the requirements statements is complete. “To be considered complete, the requirements document must exhibit three fundamental characteristics: (1) No information is left unstated or ‘to be determined’, (2) The information does not contain any undefined objects or entities, (3) No information is missing from this document” (Boehm, 1984).

The use of natural language in the definition of requirements makes validating or proving that requirements are consistent, correct and complete difficult if not impossible. “To perform consistency, completeness and correctness checking effectively and to be able to automate this process (in order to assist the requirements engineers in some of their more difficult and mundane tasks), the specification has to be expressed in a formal notation. This is because computer-based analysis requires an explicit formal semantics which provides the basis for the algorithms that carry out the analysis. This is precisely the approach that has been taken by proponents of formal methods in RE. Indeed much of the RE research effort over the last three decades has been concentrated on developing new formal requirements specification languages so that tasks such as syntax correctness, reasoning about requirements and checking their consistency can be automated in ways that are similar to how programs are compiled and managed” (Zowghi and Gervasi, 2004).
3.6.6 Key Ideas That Emerged From The Review (Research Cycle 3)

- Adopt a CAWE tool to represent the WURF process meta-model in order to resolve problems with the document based approach used in the second cycle.

- Represent completeness of requirements in a student dashboard within the CAWE tool where usage data could be captured and represented within a student dashboard. Their learning behaviour could also be tracked and analysed within the evaluation stage of the third research cycle.

- Incorporate a rules model to enforce the completeness of requirements and allow automated production of the SRS document. Consistency and correctness checking could also be performed as the student progresses through the requirements production process.

- Automated production of functional requirements including a visual representation of their relationships.

- Further address student support needs by providing guidance and examples in use within each screen within the CAWE tool. Non-textual support could also be incorporated, such as screen casts and audio feedback.

3.7 Relationship of Review to Research Cycles

Having reviewed a range of existing approaches a number of conceptual ideas emerged that provided the starting point for the design of a web user requirements approach:

3.7.1 User Modelling. Define the target audience using an appropriate model that reflects their importance using profiling and classification models.

- Primary and Secondary User Classification (UCA)
• User Profile (UCA), Person Profile (CI), Usage Scenarios / Persona (US), Actors (NDT) and Actors (UC)

3.7.2 Project Vision and Objectives. Allows the developer to establish an overall vision and business objectives before defining functional/non-functional requirements.

• Concept Vision Document (ARM), Business Vision (JAD) and Business Case (CRC)
• Business Requirements (MSF), Business Objectives (JAD)
• Requirements Generation (SSM/ICDT)

3.7.3 Task and Goal Association Model. Describe what the users do within the web/software application by the Tasks they complete or by the Goals they want to achieve. An association model links these with specific users.

• Task to Interface Object Association Model (OVID), and Tasks (CRC)(TBAS)(UC)

3.7.4 Computer Aided Web Engineering (CAWE). Automation of a rules model. Compel the student to complete every aspect of the meta-model. Check correctness of associations and consistency of requirements. Conformance to the rules model represented in the student dashboard, with visual cues to indicate completeness of the process.

• WebRatio and FlashWeb in the way it supports the developer to model aspect of the website before implementation commences.
The ‘electronic Web User Requirements Framework’ embodies work undertaken in three research cycles, where ideas evolved and changes were made in response to in class observation. Feedback from the students via in module surveys and also indirectly from delegates at the conferences that were attended all paid an important role in shaping the method and overall frameworks. It must be emphasised that research into the related work was undertaken across a period of time, as demonstrated in Volume 1, Chapter 1, section 1.7. Figure 3.8.1 shows how the review of related work maps onto the three research cycles.

Figure 3.8.1 How review of related work maps to the research cycles. A number of existing requirements process, methods and tools have been presented in this review. It is clear that from the review that: ARM; US; AMSF; and NDT address the whole requirements process, as defined by Sommerville 2007 (see Volume, Chapter 2, section 2.2.1). Five methods: NDT, AWARE, URN, SOARE and SSM/ICDT are aimed at projects that
involve Web Engineering. These are characterised by a modelling technique that enables the web developer to draw out requirements based on the business vision, objectives, the user tasks or goals and before proceeding to define both functional and non-functional requirements.

JAD, AMSF, CRC and SOARE require the development team to draw out business objectives and to link these with interactions, tasks or functional requirements. Some use a modelling approach in eliciting and defining the requirements, such as URN, AWARE and NDT. Many approaches define the user of the system as an ‘Actor’, with various methods employed to show the importance of the Actor within the system.

In the context of a web user requirements approach aimed to support the student, it is felt that techniques that use field research, which involves the collection of primary data, would result in poor adoption and usage. This is due to students having little in the way of resources (time, budget and networks) in order to realistically achieve this. One way around this is to write a briefing document, for example, as part of the ICA that contains all the key information needed to start the elicitation process. The tutor could act as the client and user within a simulation exercise within the laboratory.

The student would therefore still proceed with elicitation, without having to construct questionnaires and carry out a survey. It was found that researchers had already debated the safety of generating requirements based on “intelligent guess work” (Cato, 2001 and Szekely, 1994). This was interesting, as any approach developed from this research programme will involve making informed decisions regarding the users. Their decisions would be informed by evidence found within the ICA briefing document and as well as the tutor acting as a user. Research suggests that it is plausible to generate requirements without directly questioning the target user. “Collect facts if you
have them, or make reasonable guesses because even a reasonable guess provides a focus” (Cato, 2001). Cato encourages the use of this approach where it would prove impossible or difficult to undertake surveys to elicit requirements from the user. A technique defined by Szekely as ‘fast prototyping’, may also provide a way forward. “This approach facilitates elicitation, validation and revision through discovery of requirements. The discovery stage involves the production of a small scale version of a complicated system in order to acquire critical knowledge required to build a full system” (Szekely, 1994). This aligns with the Agile Approach to web development, involving iterative cycles of implementation and testing until the application is fit for release. Iterative development is widely used by students in the development process and is one which students would readily identify with. This method relies on revision or iterations which may prove valuable and align with the students’ development practice.

Many of the existing approaches had gaps in the treatment of user requirements, notably OVID, TBAS and CI. Web specific requirements approaches such as MSF and NDT did meet the criteria, as did ARM and US. Approaches outlined in Table 3.3 can be further characterised by:

1. Stakeholder involvement in the requirements elicitation process.
2. Detailed descriptions or profiles of the user often referred to as Actors.
3. The separation of functional and non-functional requirements.
4. Use of a meta-model to help define associations and dependencies.
5. Use of natural language to describe ‘user journeys’ or ‘scenarios’ or to map business objectives with tasks, features, behaviours and goals.
3.8 Summary

This section has reviewed, analysed and benchmarked a set of methods against key criteria developed in section 3.3. Their suitability for adoption in part, or as a whole, was also discussed. A number of ideas emerged from the review and analysis that could provide the basis of a web user requirements method to support the inexperienced student user. Mechanisms in which to resolve the ‘elicitation’ problem, where students do not have the resources to question the users directly were explored. Using simulations, briefing documents and tutors acting as the clients were all cited as a way forward.

Much of the literature on existing requirements approaches focuses on ‘software’ rather than ‘web’, pointing to a gap in knowledge in this area. RE does provide a number of important principles which need to be taken forward when thinking about the design of a web user requirements process. These include the transparency of the requirements process, where this should be logical and understood by the student together with the ability to produce valid requirements. It should also reflect agile development methods adopted by the student, including the ability to refine and append additional requirements throughout the web project period. It was argued that web requirements are distinct and require an alternative treatment within the web user requirements method to that of software focused methods.
A Framework For User Characteristics Capture: An Evaluation Of Using RUMM In The Teaching Of Web Design

Andrew Bingham
School Of Computing, University Of Teesside
a.p.bingham@tees.ac.uk

Gary Griffiths
School Of Computing, University Of Teesside
g.griffiths@tees.ac.uk

Briony Oates
School Of Computing, University Of Teesside
b.j.oates@tees.ac.uk

Julie Turnell
School Of Computing, University Of Teesside
j.d.turnell@tees.ac.uk
ABSTRACT

Students need to capture user profiles to ensure their web design projects are focused on the needs of the users. This paper outlines existing methods and proposes a new framework targeted at students on web design modules. The framework developed is named ‘Rapid User Modelling Method’ (RUMM), and acts as a cognitive aid to help students think about the target users and implications for the interface design. The framework was tested and evaluated on a group of students, who where shown to benefit from its use. In addition, the students indicated that the framework could be enhanced and developed further. The findings support the view that a method for thinking about who the user is and their requirements, can be successfully used in the teaching of web design.

Keywords


INTRODUCTION

Over the last decade web engineering has matured in terms of its approach to formalised development, starting with the adoption of information systems methodologies and progressively adopting its own methods and tools. Whilst this has led to better quality of web applications, it is believed that there is still room for improvement. The web is still in a state of massive change, with new web standards and technology pushing the need for continual professional development in web engineering. In addition there has been a paradigm shift in defining the web engineering discipline. “Web engineering is a holistic approach, and it deals with all aspects of web based systems development, starting from conception and development to implementation, performance evaluation and continual maintenance.” (Ginge And Murugesan 2001).

Within the institution a number of courses provide an opportunity for students to engage in the development of web applications using a variety of development methodologies. These include the ‘Simple Web Method’ (Lockyer, Griffiths, Hebborn, Oates, 2003) that supports the web engineering process, through a defined process of stages. Although this provides the student with a distinct process, it was found that students were not considering the user in their designs. The need to address issues of accessibility and usability within the development process has become essential, with most web development now becoming much more user focused. To address these issues, research was undertaken to support students in their analysis of the users and with a focus on interface design requirements.

This paper describes the importance of understanding the users before any design work begins, outlines existing methods for user analysis and introduces a new cognitive aid to help the student think about the user in a structured and repeatable framework. The framework was tested by 76 students during 2005/6. Students were asked to complete a survey regarding its use within the context of their projects. A proposal for further development, based upon the feedback from the student survey, is provided.
The Need For A Method To Record User Characteristics

Accessibility and the need to enhance usability within web applications has had an impact in web engineering, calling for a change in the requirements gathering phase, implementation approach and evaluation to ensure conformance to a set of pre-defined requirements. This has led to a shift towards an iterative process in web projects, where requirements conformance is measured at each stage. In addition, web projects are characterised by short development times and continual maintenance, as highlighted by Ginige and Murugesan, “Web based system development is not a one time event, as practiced by many; it’s a process with a long lifecycle.” This reinforces the notion of the web application being an evolving product, with the need for web methodologies to adopt the same approach.

The need to capture user characteristics and requirements is becoming an essential aspect of effective web engineering. A major problem that occurs in Web Engineering projects is that the users get to know how to express their requirements very late in the process, i.e. after the design artefacts appeared. (Lowe and Eklund, 2002) To address these issues, there has been a move towards more user centered methods that have a user requirements stage explicitly integrated into the process. It is recognised that while this does offer a more user focused product, the ‘user centered design’ ethos needs to go beyond this. “The need for a systematic approach to capture user navigational requirements has some merit and perhaps some urgency.” (Barry and Lang, 2001)

User analysis for a commercial web project can be undertaken using a variety of techniques, including questionnaires and focus groups. The data could then be used to influence subsequent stages in the development process. Within the context of an educational web project, questionnaires are often an unrealistic proposition for students due to cost and time implications. Whilst this constraint is a major barrier to conducting user analysis, it was still felt that a tool or framework was still required to enable the student to think about users characteristics and produce a set of guidelines or requirements for the design of a web interface.

Before any user requirements can be documented, a method to capture their requirements must be used. Existing approaches include;

1. Concur Task Trees. Allows the envisioning user interaction with the application and derive information, navigation and presentation design accordingly. (Bolchini, Mylopoulos, 2003)

2. Task Based Audience Segmentation. A technique that defines the target audience by the tasks they perform to achieve a goal. (Young, 2005)

3. Ethnographic Approach / Contextual Enquiry. Provides a framework that helps understand users and their requirements using a structured interview. (Gerry Gaffney, 2004)

Construction Of The Method

The main aim of the method is to ensure that the student thinks about who the users are and the subsequent implications for the design of an interface, before any design work starts. It was also important that the framework should integrate with existing methodologies, or indeed be used in isolation to ensure adoption amongst a diverse student population.
Having established the need for students to think more deeply about the users, the next step was to outline the objectives of the framework.

1. The process must be rapid, without the need to collect primary data.
2. The student must be able to identify and understand the language used.
3. Should be accessible to students who are by definition less experienced.
4. To communicate a set of requirements for interface design.
5. It should consider the notion of multiple users of websites, rather than one.

Current practice in web development considers the identification of the user profile as the starting point of the user requirements analysis (Bolchini & Eric, 2004). A user profile describes stable archetypal qualities of a relevant target segment (Carroll 2002) and may comprise a variety of attributes based on demographic eg. age, gender, occupation, disabilities etc. or “webographic” eg. net usage habits, interests and software constraints, favourite sites etc. (Garrett, 2002). Profiles can be discovered through a variety of requirements elicitation techniques based on user research, such as surveys, contextual inquiry, focus groups and structured interviews. (Bolchini et al., 2004). The user profile can be broken down into four elements; a person profile, a web use profile, a context of use profile and personalising the user. The profile should be as accurate, clear and realistic as possible.

The mechanism in which the student obtains the data about the user is crucial. In the context of their projects, it was felt that techniques that use primary data would result in poor adoption of the framework. This is due to time and cost implications that would be placed on the student. As such, the framework must use an alternative method to generate the profile of the users. Any such method would rely upon judgment and reasonable guesses from the student themselves. “Collect facts if you have them, or make reasonable guesses because even a reasonable guess provides a focus”. (Cato, 2001). Although it could be argued that this would result in an un-safe profile, the student can obtain some limited information about the user. This can found in ICA briefing documents where the user is loosely defined, but where much more analysis is expected.

Having undertaken the background research into existing methods for creating user profiles and more general user analysis, it was felt that a new cognitive aid and frame was required. The research would help shape the framework in terms of characteristic capture and how this could be communicated and synthesised into a set of requirements for the interface design.

Making the student think about ‘who’ the users are and how they would use the application would be the main objective. The framework would rely upon decision making from the student themselves, since no actual data would be collected. It must be emphasized that RUMM must be completed by students themselves, in the context of a project that they are working on. Subsequently the only reasonable outcome of the system would be merely to make the student think, rather than automatically produce the requirements.

The following questions provided a basis for the full development of RUMM;

1. Who are the users?
(age/gender/culture)

2. What do the users expect to do with the application?
(To learn, purchase online, assist them in their job, provide fun or leisure activity).

3. When will the users use the application?
(In the course of their job, in library or other access point, at home in their leisure time).

4. How will the users use the application?
(At home/work, on a modem connection, broadband, cd-rom, PC/Mac/Linux, Kiosk, PDA and with assistive technologies).

Once the student has thought about the characteristics of the users and how they intend to use the application, they are then expected to write a description of the Primary and Secondary user. By providing a framework for the student to think about the users, it was hoped that a much more detailed and focused description could be developed for the project. After the description is developed, they must think about the considerations for the layout, colour, content and navigation of the user interface. It is at this stage that the student would form the requirements for the interface design.

To view the RUMM document please refer to appendix 1.0.

Results Of The Student Survey

At the end of the module students were asked to complete a questionnaire regarding their opinion of using RUMM. These were returned anonymously to ensure an un-biased response. Seventy six students were asked to complete the questionnaire on a final year degree and on a postgraduate module. Fifteen questionnaires were returned by the undergraduates and six questionnaires were returned by the postgraduates. The response rate of 16% is considered to be normal for a questionnaire survey. The questionnaires asked the students if it helped them define users for their project, whether there were any elements missing that they felt should be included or if any unnecessary information was included. Additionally students were asked if they would use a more advanced tool based on the RUMM framework. To view the questionnaire used in the survey please refer to appendix 2.0.

Twenty students (95%) expressed that they found RUMM ‘very useful/useful’ in defining the audience for the projects that they were working on. When asked if they had used other approaches for defining the user 4 students (19%) said that they had never used other approaches, and of those who had (62%), said that it was more useful compared to the approaches used previously. Four students (19%) expressed the opinion that it was not much better then an approach they had used in the past.

The results from this section of the survey are very encouraging in terms of acceptance of the framework from the students. Although the framework is still at a very early stage of development, students felt that it did help them to define who the user is and a significant number of students felt RUMM was more useful than approaches they had used before. An
interesting outcome of question two was the fact that students had been using previous approaches. Whilst this is true, the definition of an approach is critical here, as this had an ambiguous meaning. An approach could be defined as a rigid framework or conversely a more loose process of simply writing a paragraph about the user. The latter fits in well with the evidence available in many student in-course assessments (ICA) where they had not used RUMM and the user was only briefly considered.

Question three asked if there were any points missing from RUMM that should be considered. The responses were quite varied and include:

More Focus on personal background of users, including profession, hobbies and customs to help understand them better.

Non-native English speakers (specify level), users reading from right to left or from bottom to top, disabled users with/without assistive technologies, use at school/college/university.

Level of computer use/competence section, we need to know what is actually meant by novice, intermediate, expert.

This feedback is again quite interesting in that the students want to break down the characteristics even further. Some students felt that more characteristics should be integrated into RUMM, such as hobbies and the profession. It could be argued that these are perhaps too detailed and specific but are none the less, important in thinking about the user. Whilst it is felt these characteristics are valid, there might be a better way of thinking about the user at this level of detail. This is discussed in more detail in next section of this paper.

Question four asked if they felt that there was any unnecessary information being gathered through the method. Only one student responded to this question, where they thought there should be a clearer way of differentiating primary users from secondary users. It was felt that this is a very important aspect of RUMM that could have implications for the way in which a priority system could be integrated to ensure no conflicts arise.

Question five asked the student if an enhanced version of RUMM which produced design guidelines was available for them to use, would they find this useful. Thirteen students (62%) said that they would find such a tool very useful, seven student (33%) said that they would find it useful.

**Evaluation & Further Development**

The results of the survey reinforce the positive impact that RUMM has had on teaching. Whilst the framework is still in early development, it is evident that students who have used it are confident in its application. One area for improvement would be the second half of RUMM, where they have to write a brief summary of the primary, secondary users and think about the navigation, content, layout and colour. During this part of RUMM, reliance is placed on the knowledge of the student and there is a preconception that they understand usability and conventions associated with the application of colour and layout in interface design. This stage of RUMM requires more investigation, especially pertaining to the safety of assumptions made by
the students. The approach of using characteristics to model the user is central to the framework, but it is recognised that this can be difficult for the student.

One area which may help the student identify more with the process, is the use of persona’s. RUMM could be used to generate a number of different persona’s, based on a predefined model. For example the use of persona’s could help provide a rich contextual model of a group of primary targeted users based on conjoint analysis theory. The persona analysis process is divided into the two processes of simultaneous disjoint clustering, and identification of personas. These persona’s could be clustered to generate standardised requirements, based upon a minimal set of the most influential users. (Mikio Aoyama, 2005)

The framework thus so far has concentrated on user characteristics. But users also exhibit other traits such as motivation and needs. It is felt these are features lacking in the current version of RUMM, and as such more development is needed to consider their integration. In addition, further work will explore the traceability of RUMM to evaluate its influence on the design process. This would involve the investigation of the application of profiles upon a web site to determine differences in design approaches using a test project.

The use of RUMM has had a positive impact on the way students think about the user, and we believe the main objective of the framework has been realised. Once the tool had been used by students, some weaknesses became apparent. These mainly related to the use of primary and secondary users and dealing with conflicts and priorities which may arise. Additionally we had expected the students to understand the language used to define the characteristics, an assumption that we perhaps should have first tested in a pilot of the framework.

Conclusions
This paper has established that there is a need to encourage students to think more deeply about who the users are, what they do with the application, when and how they use it. There are very few methods that can achieve this without resorting to expensive and time consuming techniques, and which would not fit with the needs of the students themselves. The majority of students indicated that they found RUMM useful in helping them define the users for their projects. They also felt that there needs to be some mechanism to automate the requirements for layout, content, navigation and colours, perhaps via the production of guidelines. User characteristics could be expanded further, with additional components and the modification of others.

It is hoped that the next cycle of research would build upon the positive elements of the RUMM framework, and test other approaches to enhance the process. One of the ways in which this could be achieved is the integration of persona’s and clustering. The clustering technique itself would require more research, in addition to investigating how to turn the outcomes into usable guidelines.

When we first investigated the need for a better approach for thinking about the user, we set out our objectives for the framework. These included the need for a rapid approach, helping the student to communicate requirements through a simple framework that they could use without any previous experience. It should also reinforce the notion of multiple users. The investigation
and subsequent development of the RUMM framework showed that most objectives were attained and that RUMM provides a basis for the evolution of a more advanced tool.

Acknowledgements

We would like to thank the students in the School of Computing, University of Teesside who were willing to use RUMM during their projects and answer our survey during a very busy period.

REFERENCES


Determing Web User Requirements Through User Modelling: A Framework For User Characteristics Capture

Andrew Bingham

School of Computing, University of Teesside, Borough Rd, Middlesbrough, TS1 3BA, UK.

Abstract
The research explores the need to capture user requirements in the context of web engineering projects. The objective of the research is to develop a cognitive aid and framework. The author argues that the need for such a framework exists because of the changing nature of web development, with time to market pressures for developers, and a fluid and constantly changing and evolving user. The first cycle of research has been completed and evaluated. The evidence suggests that such a framework benefits students when working on web development projects.

Keywords (User Characteristics, User Analysis, Evaluation, method-in-action, Persona, Web Design)

1. Introduction
Over the last decade web engineering has matured in terms of its approach to formalised development, starting with the adoption of information systems methodologies and progressively adopting its own methods and tools. Whilst this has led to better quality of web applications, it is believed that there is still room for improvement. The web is still in a state of massive change, with new web standards and technology pushing the need for continual professional development in web engineering. In addition there has been a paradigm shift in defining the web engineering discipline. “Web engineering is a holistic approach, and it deals with all aspects of web based systems development, starting from
conception and development to implementation, performance evaluation and continual maintenance.” (Ginge And Murugesan 2001).

Within the school of computing, a number of courses provide an opportunity for students to engage in the development of web applications using a variety of development methodologies. These include the ‘Simple Web Method’ (M A Lockyer, G Griffiths, B D Hebbron, B J Oates, 2003) that supports the web engineering process, through a defined process of stages. Although this provides the student with a distinct process, it was found that students were not considering the user in their designs. The need to address issues of accessibility and usability within the development process has become essential, with most web development now becoming much more user focused. To address these issues, research was undertaken to support students in their analysis of the users within the context of the project. The term ‘users’

This paper describes the importance of the research within the web engineering discipline and establishes the position the framework will take within the development lifecycle. It will outline the proposed research paradigm and propose an appropriate methodical research approach to be taken by the researcher.

2. The Need For A Method To Record User Characteristics

Accessibility and the need to enhance usability within web applications has had an impact in web engineering, calling for a change in the requirements gathering phase, implementation approach and evaluation to ensure conformance to a set of pre-defined requirements. This has led to a shift towards an iterative process in web projects, where requirements conformance is measured at each stage. In addition, web projects are characterised by short development times and continual maintenance, as highlighted by Ginige and Murugesan, “Web based system development is not a one time event, as practiced by many; it’s a process with a long lifecycle.” This reinforces the notion of the web application being an evolving product, with the need for web methodologies to adopt the same approach.

The need to capture user characteristics and requirements is becoming an essential aspect of effective web engineering. A major problem that occurs in Web Engineering projects is that the users get to know how to express their requirements very late in the process, i.e. after the design artefacts appeared. (Lowe and Eklund, 2002) To address these issues, there has been a move towards more user centered methods that have a user requirements stage explicitly integrated into the process. It is recognised that while this does offer a more user focused product, the ‘user centered design’ ethos needs to go
beyond this. “The need for a systematic approach to capture user navigational requirements has some merit and perhaps some urgency.” (Barry and Lang, 2001)

User analysis for a commercial web project can be undertaken using a variety of techniques, including questionnaires and focus groups. The data could then be used to influence subsequent stages in the development process. Within the context of an educational web project, questionnaires are often an unrealistic proposition for students due to cost and time implications. Whilst this constraint is a major barrier to conducting user analysis, it was still felt that a tool or framework was required to enable the student to think about users characteristics and produce a set of guidelines or requirements for the design of a web interface.

Students need to consider the role of the user in their web development projects to ensure embedded artefacts are focused on the needs of the users. Usually the student is using an existing methodology or creating their own hybrid to best fit the project. As such it is essential that the framework can be used with a variety of development methodologies or indeed used in isolation.

Before any user requirements can be documented, a method to capture their requirements must be used. Existing approaches include;

1. **Concur Task Trees.** Allows the envisioning user interaction with the application and derive information, navigation and presentation design accordingly. (Bolchini, Mylopoulos, 2003)

2. **Task Based Audience Segmentation.** A technique that defines the target audience by the tasks they perform to achieve a goal. (Young, 2005)

3. **Ethnographic Approach / Contextual Enquiry.** Provides a framework that helps understand users and their requirements using a structured interview. (Gerry Gaffney, 2004)

### 3. Construction Of The Method
The main aim of the method is to ensure that the student thinks about who the users are and the subsequent implications for the design of an interface, before any design work starts. It was also important that the framework should integrate with existing methodologies, or indeed be used in isolation to ensure adoption amongst a diverse student population.
Having established the need for students to think more deeply about the users, the next step was to outline the objectives of the framework.

1. The process must be rapid, without the need to collect primary data.
2. The student must be able to identify and understand the language used.
3. Should be accessible to students who are by definition less experienced.
4. To communicate a set of requirements for interface design.
5. It should consider the notion of multiple users of websites, rather than one.

Current practice in web development considers the identification of the user profile as the starting point of the user requirements analysis (Cato, 2001). A user profile describes stable archetypal qualities of a relevant target segment (Carroll, 2002) and may comprise a variety of attributes based on demographic eg. age, gender, occupation, disabilities etc. or “webographic” eg. net usage habits, interests and software constraints, favourite sites etc. (Garrett, 2002). Profiles can be discovered through a variety of requirements elicitation techniques based on user research, such as surveys, contextual inquiry, focus groups and structured interviews. (Bolchini & Eric, 2004).

The mechanism in which the student obtains the data about the user is crucial. In the context of their projects, it was felt that techniques that use primary data would result in poor adoption of the framework. This is due to time and cost implications that would be placed on the student. As such, the framework must use an alternative method to generate the profile of the users. Although it could be argued that this would result in an un-safe profile, the student can obtain some limited information about the user. This can found in ICA briefing documents where the user is loosely defined, but where much more analysis is expected.

Having undertaken the background research into existing methods for creating user profiles and more general user analysis, it was felt that a new student centered method was required. The research would help shape the framework in terms of characteristic capture and how this could be communicated and synthesised into a set of requirements for the interface design.

Making the student think about ‘who’ the users are and how they would use the application would be the main objective. The framework would rely upon decision making from the student themselves, since no actual data would be collected. Subsequently the
only reasonable outcome of the system would be merely to make the student think, rather than automatically produce the requirements.

The following questions provided a basis for the full development of RUMM;

- **Who are the users?** (age/gender/culture)

- **What do the users expect to do with the application?** (To learn, purchase online, assist them in their job, provide fun or leisure activity).

- **When will the users use the application?** (In the course of their job, in library or other access point, at home in their leisure time).

- **How will the users use the application?** (At home/work, on a modem connection, broadband, cd-rom, PC/Mac/Linux, Kiosk, PDA and with assistive technologies).

Once the student has thought about the characteristics of the users and how they intend to use the application, they are then expected to write a description of the **Primary** and **Secondary** user. By providing a framework for the student to think about the users, it was hoped that a much more detailed and focused description could be developed for the project. After the description is developed, they must think about the considerations for the layout, colour, content and navigation of the user interface. It is at this stage that the student would form the requirements for the interface design.

### 4. Results Of The Student Survey

At the end of the module students were asked to complete a questionnaire regarding their opinion of using RUMM. These were returned anonymously to ensure an un-biased response. Seventy six students were asked to complete the questionnaire on a final year degree and on a postgraduate module. Fifteen questionnaires were returned by the undergraduates and six questionnaires were returned by the postgraduates. The questionnaires asked the students if it helped them define users for their project, whether there were any elements missing that they felt should be included or if any unnecessary information was included. Additionally students were asked if they would use a more advanced tool based on the RUMM framework. Please see **appendix 2.0** for a copy of the questionnaire.

Twenty students (95%) expressed that they found RUMM ‘very useful/useful’ in defining the audience for the projects that they were working on. When asked if they had used other approaches for defining the user 4 students (19%) said that they had never used other approaches, and of those who had (62%), said that it was more useful compared to
the approaches used previously. Four students (19%) expressed the opinion that it was not much better than an approach they had used in the past.

The results from this section of the survey are very encouraging in terms of acceptance of the framework from the students. Although the framework is still at a very early stage of development, students felt that it did help them to define who the user is and a significant number of students felt RUMM was more useful than approaches that they had used. An interesting outcome of question two was the fact that students had been using previous approaches. Whilst this is true, the definition of an approach is critical here, as this had an ambiguous meaning. An approach could be defined as a rigid framework or conversely a more loose process of simply writing a paragraph about the user. The latter fits in well with the evidence available in many student ICA’s where they had not used RUMM and the user was only briefly considered.

Question three asked if there were any points missing from RUMM that should be considered. The responses were quite varied and include;

- More Focus on personal background of users, including profession, hobbies and customs to help understand them better.
- Non-native English speakers (specify level), users reading from right to left or from bottom to top, disabled users with/without assistive technologies, use at school/college/university.
- Level of computer use/competence section, we need to know what is actually meant by novice, intermediate, expert.

This feedback is again quite interesting in that the students want to break down the characteristics even further. Some students felt that more characteristics should be integrated into RUMM, such as hobbies and the profession. It could be argued that these are perhaps too detailed and specific but are none the less, important in thinking about the user. Whilst it is felt these characteristics are valid, there might be a better way of thinking about the user at this level of detail. This is discussed in more detail in section five of this paper.

Question four asked if they felt that there was any unnecessary information being gathered through the method. Only one student responded to this question, where they thought there should be a clearer way of differentiating primary users from secondary
users. It was felt that this is a very important aspect of RUMM that could have implications for the way in which a priority system could be integrated to ensure no conflicts arise.

Question five asked the student if an enhanced version of RUMM which produced design guidelines was available for them to use, would they find this useful. Thirteen students (62%) said that they would find such a tool very useful, seven student (33%) said that they would find it useful.

5. Evaluation & Further Development

The results of the survey reinforce the positive impact that RUMM has had on teaching. Whilst the framework is still in early development, it is evident that students who have used it are confident in its application. One area for improvement would be the second half of RUMM, where they have to write a brief summary of the primary, secondary users and think about the navigation, content, layout and colour. During this part of RUMM, reliance is placed on the knowledge of the student and there is a preconception that they understand usability and conventions associated with the application of colour and layout in interface design. This stage of RUMM requires more investigation, especially pertaining to the safety of assumptions made by the students. The approach of using characteristics to model the user is central to the framework, but it is recognised that this can be difficult for the student.

One area which may help the student identify more with the process, is the use of persona’s. RUMM could be used to generate a number of different persona’s, based on a predefined model. For example the use of persona’s could help provide a rich contextual model of a group of primary targeted users based on conjoint analysis theory. The persona analysis process is divided into the two processes of simultaneous disjoint clustering, and identification of personas. These persona’s could be clustered to generate standardised requirements, based upon a minimal set of the most influential users. (Mikio Aoyama, 2005)

The framework thus so far has concentrated on user characteristics. But users also exhibit other traits such as motivation and needs. It is felt these are features lacking in the current version of RUMM, and as such more development is needed to consider their integration.

The use of RUMM has had a positive impact on the way students think about the user, and we believe the main objective of the framework has been realised. Once the tool had been used by students, some weaknesses became apparent. These mainly related to the use of primary and secondary users and dealing with conflicts and priorities which may arise. Additionally we had expected the students to understand the language used to
define the characteristics, an assumption that we perhaps should have first tested in a pilot of the framework.

6. Conclusions
The need to think about users before designing an interface is important. There are very few methods that can achieve this without resorting to expensive and time consuming techniques. The majority of students indicated that they found RUMM useful in helping them define the users for their projects. They also felt that there needs to be some mechanism to automate the requirements for layout, content, navigation and colours, perhaps via the production of guidelines. User characteristics could be expanded further, with additional components and the modification of others.

It is hoped that the next cycle of research would build upon the positive elements of the RUMM framework, and test other approaches to enhance the process. One of the ways which this could be achieved is the integration of persona's and clustering. The clustering technique itself would require more research, in addition to investigating how to turn the outcomes into usable guidelines.

When we first investigated the need for a better approach for thinking about the user, we set out our objectives for the framework. These included the need for a rapid approach, helping the student to communicate requirements through a simple framework that they could use without any previous experience. It should also reinforce the notion of multiple users. The investigation and subsequent development of the RUMM framework showed that most objectives were attained and that RUMM provides good basis for the evolution for a more advanced tool.

References


Appendix A5 - Online Survey Questionnaire

Question 1. Please indicate which course you are currently enrolled.

BSc International Business Information Technology

BSc Computer Studies

BA Web Design

BA Creative Digital Media

BSc Web Development

BSc IT and Networks

Creative Multimedia

Masters Course

Other

Question 2. Please Indicate Your Mode Of Study

Full-time:

Part-time:

Question 3. Did you use WURF in your in-course assessment

Yes:

No: If no, please move onto Question 10.

Now thinking about how you used WURF in your in-course assessment (ICA). The next four statements use a ‘Likert Scale’ rating scheme based on the suitability of WURF for performing the requirements analysis.

The scale key is as follows:
1 = Strongly Agree 2 = Agree 3 = Neither agree or disagree 4 = Disagree 5 = Strongly Disagree

Statement 4a. I understood the process of WURF without the need to ask for help.

- 1
- 2
- 3
- 4
- 5

Statement 4b. WURF helped me think about the user in terms of characteristics and their requirements within the web application.

- 1
- 2
- 3
- 4
- 5

Statement 4c. WURF helped me think about translating requirements into tasks and functions within the application.

- 1
- 2
- 3
- 4
- 5

Statement 4d. WURF takes too much time to complete.

- 1
- 2
- 3
- 4
- 5

Statement 4e. There’s no benefit for me in using WURF.
Statement 4f. Before using WURF I had not thought enough about the user and their requirements.

Question 5. What step within WURF did you least understand?
   a. Statement Of Purpose / Vision / Define Web Application Objectives
   b. Define Web Actors / Define Web Actor Tasks
   c. Define Web Functional Requirements
   d. Define Web Non-Functional Requirements

Question 6. Which aspect of WURF did you feel helped the most?
   a. Statement Of Purpose / Vision / Define Web Application Objectives
   b. Define Web Actors / Define Web Actor Tasks
   c. Define Web Functional Requirements
   d. Define Web Non-Functional Requirements

Question 7. Would You Use WURF again?
   Yes:
   No:
Question 8. Now thinking about the ‘user characteristic’ stage. Would you prefer to use ‘pre-written’ persona’s / profiles here rather than writing these yourself?

Yes: ●

No: ●

I Don’t Understand this question: ●

Question 9. Are there any additional enhancements that you would like to see incorporated? Please indicate these in the box below:

Submit Questionnaire

Question 10. Have you used another method for User Requirements Analysis?

Yes: (please indicate this the text feild below) ●

No: ●

if Yes, please write here which one you used.
<table>
<thead>
<tr>
<th>Course</th>
<th>Mode</th>
<th>Use WURF in ICA</th>
<th>4a</th>
<th>4b</th>
<th>4c</th>
<th>4d</th>
<th>4e</th>
<th>4f</th>
<th>What step within WURF did you least understand?</th>
<th>Which aspect of WURF did you feel helped the most?</th>
<th>Use again?</th>
<th>Enhancements</th>
<th>Analysis?</th>
<th>Note</th>
<th>Have You used another method for requirements analysis?</th>
<th>RUMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Masters</td>
<td>Full-time</td>
<td>yes</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>Define Web Non-Functional Requirements</td>
<td>Statement Of Purpose / Vision / Define Web Applica</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 ComputerStudies</td>
<td>Part-time</td>
<td>yes</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>Define Web Non-Functional Requirements</td>
<td>Define Web Actors / Define Web Actor Tasks</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 WebDesign</td>
<td>Full-time</td>
<td>yes</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>Define Web Non-Functional Requirements</td>
<td>Define Web Actors / Define Web Actor Tasks</td>
<td>yes</td>
<td>no</td>
<td>yes RUMM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 ComputerStudies</td>
<td>Full-time</td>
<td>yes</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>Define Web Non-Functional Requirements</td>
<td>Statement Of Purpose / Vision / Define Web Applica</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 ComputerStudies</td>
<td>Full-time</td>
<td>yes</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>Define Web Actors / Define Web Actor Tasks</td>
<td>Statement Of Purpose / Vision / Define Web Applica</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 ComputerStudies</td>
<td>Part-time</td>
<td>yes</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>Define Web Actors / Define Web Actor Tasks</td>
<td>Statement Of Purpose / Vision / Define Web Applica</td>
<td>yes</td>
<td>no</td>
<td>n/a RUMM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 CreativeMultimedia</td>
<td>Full-time</td>
<td>yes</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>Statement Of Purpose / Vision / Define Web Applica</td>
<td>Statement Of Purpose / Vision / Define Web Applica</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 ComputerStudies</td>
<td>Part-time</td>
<td>yes</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>Define Web Actors / Define Web Actor Tasks</td>
<td>yes no N/A RUMM V1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 ComputerStudies</td>
<td>Part-time</td>
<td>yes</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>Define Web Actors / Define Web Actor Tasks</td>
<td>Statement Of Purpose / Vision / Define Web Applica</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Masters</td>
<td>Full-time</td>
<td>yes</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>Define Web Non-Functional Requirements</td>
<td>Statement Of Purpose / Vision / Define Web Applica</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11 IBIT</td>
<td>Full-time</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>yes</td>
<td>RUMM2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12 Masters</td>
<td>Full-time</td>
<td>yes</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>Define Web Non-Functional Requirements</td>
<td>Define Web Actors / Define Web Actor Tasks</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13 WebDesign</td>
<td>Full-time</td>
<td>yes</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>Define Web Non-Functional Requirements</td>
<td>Define Web Actors / Define Web Actor Tasks</td>
<td>yes</td>
<td>no</td>
<td>yes RUMM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14 WebDesign</td>
<td>Full-time</td>
<td>yes</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>Define Web Functional Requirements</td>
<td>Statement Of Purpose / Vision / Define Web Applica</td>
<td>yes</td>
<td>no</td>
<td>yes RUMM</td>
<td></td>
<td>RUMM, questionnaire, personas</td>
<td></td>
</tr>
<tr>
<td>15 CreativeMultimedia</td>
<td>Full-time</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>yes</td>
<td>RUMM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 WebDesign</td>
<td>Full-time</td>
<td>yes</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>Define Web Non-Functional Requirements</td>
<td>Statement Of Purpose / Vision / Define Web Applica</td>
<td>yes</td>
<td>no</td>
<td>yes rumm</td>
<td></td>
<td>rumm version 1</td>
<td></td>
</tr>
<tr>
<td>17 CreativeMultimedia</td>
<td>Full-time</td>
<td>yes</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>Define Web Non-Functional Requirements</td>
<td>Define Web Functional Requirements</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix B1 - CAWE Development and Implementation

Development Environment

A number of options were considered for the implementation of the CAWE tool. These were all centred upon contemporary ‘web application development platforms’ governed by the need to integrate with existing systems such as the institutions VLE, intranet and to store user data. The researchers own skill set was also taken into consideration. Standalone platforms that had to be installed on a PC were considered, for example, Adobe Integrated Runtime (AIR) was a platform that could be used, as it provides an installation package that can be distributed via the VLE. AIR can also access user data via a web service for the purposes of consuming user data. This was an important consideration given the need to access and update data generated within CAWE tool. PHP and MySQL could then used to control the rules model, but would require the data to be transformed into a web service for consumption by the AIR application. This adds a layer of complexity and additional overhead in terms of time in order to develop and test. Alternatively, a browser based solution using XHTML and CSS would offer a number benefits:

- The author has ten years experience of developing XHTML, CSS, PHP and MySQL web applications and has a number of pre built libraries that could be utilised.

- Any problems with the CAWE tool can be updated and rolled out instantly using a browser based web application. If AIR was adopted, the application would need to be re-installed on each machine.

- It would not be necessary to build a web service for AIR to consume user data.

The CAWE tool is as a mission critical application in terms of this research programme and for its use within the module curricular. Using new
technologies such as AIR may provide benefits from a student experience perspective, but with an additional overhead in respect of time to learn. A pragmatic approach to development was needed due to the very short development lifecycle, as design and development time was limited to two months in total. It was decided to use XHTML, CSS, PHP and MySQL as the development environment, which would expedite the release of the CAWE tool in time for the new term.

**Analysis and Design**

The existing meta-model was modified and transformed into a class diagram in order to address the issues highlighted in the evaluation, as discussed in the previous section. A rules model had to be written to control a number of aspects of the CAWE tool, including the consistency, completeness and correctness and checking of requirements and the student dashboard. This is represented within the ‘Dashboard class’ as CompletenessAndConsistency(), a core function within the CAWE tool that drives the student dashboard. The dashboard is dynamically created, based on the completeness of the requirements and provides feedback to the student regarding which aspects need to be completed next.

The CompletenessAndConsistency() function is dependent on the TaskToActorAssoc() and FunctionToActorAssoc() functions. In the second cycle, it was identified that being able to associate tasks and functional requirements to specific actors would enhance the traceability aspect of WURF, but was something that could not be achieved correctly in the paper based tool. The CAWE tool provides an opportunity to represent and control associations more precisely and impose a set of rules to govern how these are created and subsequently modified. For example, if an Web Actor is deleted part way through the project, the dependenciesCheck() function within the webAppActor class is instantiated and provides feedback within the student dashboard. Once the student confirms that they wish to remove the actor, the association functions are instantiated to remove the actor in all association instances.

One of the main benefits to the student in using the CAWE tool is the ability to automatically generate an SRS document, which has version control to ensure they can revert to a previous version if required. This necessitated the use of two classes, RequirementDocument and RequirementPattern, each with
its own set of functions to transform user data into an SRS. The latter class is required for the generation of a ‘virtual document’, held in a separate data store for later retrieval by the RequirementDocument class. This ‘virtual document’ ensures that each SRS is unique for version control purposes and so that unique identifiers can be faithfully reproduced in each document for traceability purposes.

Figure 8.2.3,(see Volume 1, page 224), illustrates all the classes within the CAWE tool, including the consumption of data outside the scope of the application, represented in the ‘module class’. These provided the basis for the implementation and transformation into PHP classes within the CAWE tool.

CAWE Tool Implementation

A suitable Integrated Development Environment (IDE) was chosen based upon the need to undertake both client-side and server-side scripting and to test key classes during development. Dreamweaver was chosen due to its ability to achieve this and the way it was able to provide a mechanism to write valid PHP classes. Rapid Prototyping was used to develop the CAWE tool, reflecting the short development time available. A total of nine iterations within the rapid prototyping approach provided stable release versions. Classes were written and immediately tested on a live server environment, rather than on a local server. This ensured syntax and logic errors could be quickly identified, rectified and re-tested to ensure conformity to the model. Error reporting was switched on within the PHP configuration settings to help identify syntax errors.

A client-side HTML template was produced, (see Figure 6.2.5), using both XHTML and CSS that included generic content holders to enable dynamic server-side functions to be embedded later. The class diagram provided the basis for transformation of the model into PHP classes. A simple version control system was adopted, with the benefit of being able to return to a previous working version if required.
Figure B1.1 Client-side HTML template.

The starting point for the development was dictated by the need to solve the most difficult problems first. The author had already implemented a library of authentication, database connectivity, user data (adding, editing and deleting) classes and so these could be integrated easily when required. It was identified that the ‘Web Actor to Task’ association class was the most challenging aspect of the implementation phase due to the multi-dimensional relationships between both Task/Functional Requirements Web Actors. The rules model had to be represented within the application logic. For example, the dependenciescheck() function had to be able to access relationships between Tasks and Web Actors. Any routine must be able to check for changes to multiple dependencies and be able to report these to the student dashboard. Accordingly, the TaskToActorAssoc() and FunctionToActorAssoc() association classes were chosen as the starting point for the first iteration within the implementation phase.
Association Classes – First Iteration

Initial ideas regarding the association class included the ability to relate a maximum number of four actors to any one task or functional requirement. The rules model imposed certain constraints on this process, for example, in order to reflect changing tasks and functional requirements, dependencies checking must take place. If an actor was no longer required but was associated to a task or functional requirement, this should be indicated within the student dashboard during the deletion routine.

Before the association routine could be written, it was important that a limited number of classes were written and tested, as both the TaskToActorAssoc() and FunctionToActorAssoc() classes consumed pre-existing data. The following classes formed the basis of the first iteration; WebAppActor, WebAppTask, WebAppFunction (see Figure B1.2).
An association between individual Web Actors and Tasks had to be modelled, with the routine that enabled the student to *add*, *edit* and *delete* both Tasks and Web Actors in later versions of their requirements specifications. It was therefore essential that dependencies could be checked and edited by the `dependenciesCheck()` function without recourse to elaborate SQL queries. Early attempts included limiting the association data structure to a maximum of four actors. Each association would have a unique id (TaskToActorAssoc_ID), a foreign key (WebAppTask_ID) and names of the actors for the association. In order to check ownership of the association, an Owner_ID key was included (see Figure B1.3).

<table>
<thead>
<tr>
<th>TaskToActorAssoc</th>
<th>WebAppActor</th>
<th>WebAppTask</th>
</tr>
</thead>
<tbody>
<tr>
<td>TaskToActorAssoc_ID</td>
<td>WebAppTask_ID</td>
<td>Owner_ID</td>
</tr>
<tr>
<td>WebAppActor_1</td>
<td>taskName</td>
<td>Owner_ID</td>
</tr>
<tr>
<td>WebAppActor_2</td>
<td></td>
<td>Add() Edit() Delete()</td>
</tr>
<tr>
<td>WebAppActor_3</td>
<td></td>
<td>dependenciesCheck()</td>
</tr>
<tr>
<td>WebAppActor_4</td>
<td></td>
<td>MakeAssociation()</td>
</tr>
<tr>
<td>Owner_ID</td>
<td></td>
<td>EditAssociation()</td>
</tr>
</tbody>
</table>

Figure B1.3 TaskToActor Data Structure in prototype 1.

The first attempt was partially successful, in that it was possible to create the record for the association, but it was recognised that two issues would become a barrier for further progression:

1. It was only possible to associate four actors.
2. Data integrity problems. To address this, the primary key of the WebAppActor table should have been used.
In particular, the second problem regarding data integrity issues would have meant writing complex SQL to retrieve the correct data for the actor id association and would result in explicitly relating both the TaskToActorAssoc and WebAppActor when the student edits an actor. The first problem would prevent extension of the tool to include an unlimited number of associations. In addition, the student was presented with a limited number of actors and tasks in their dashboard screen.

**Association Class – Second Iteration**

In order to resolve some of the issues presented in the first iteration, attention was paid to the student’s ‘Define Web Actor’ dashboard screen. In the first iteration this was ‘hard coded’ to a maximum of ten actors which proved to be too restrictive. The screen was redeveloped to include a feature that enabled the student to add unlimited actors. This was achieved by inclusion of a button ‘Add Another Actor’, written in javascript (see Figure B1.4).

![Figure B1.4 Add Web Actors Screen with ‘Add Another Actor’ button.](image)

The TaskToActorAssoc class structure was modified to enable a one-to-four (one task-to-four actors) association model. An advantage of this was the ability to model relationships using the primary key of the actors table, rather
than the name of the actor. If an actor name was changed, the integrity of its relationship is maintained and preservation of the association remains (see Figure B1.5). Actor associations were limited within this iteration, but the approach offered a way forward for one-to-many relationships.

<table>
<thead>
<tr>
<th>TaskToActorAssoc</th>
</tr>
</thead>
<tbody>
<tr>
<td>TaskToActorAssoc_ID</td>
</tr>
<tr>
<td>WebAppTask_ID</td>
</tr>
<tr>
<td>WebAppActor_ID</td>
</tr>
<tr>
<td>Owner_ID</td>
</tr>
<tr>
<td>MakeAssociation()</td>
</tr>
<tr>
<td>EditAssociation()</td>
</tr>
</tbody>
</table>

**Figure B1.5 Modified TaskToActorAssoc class.**

The ‘Task To Actor’ association dashboard was then developed to enable the student to select specific actors to be associated with a particular task. A routine was developed to list each task in ascending order, which then selected actor data from the WebAppActor table. Four dropdown menus were dynamically created next to each task (see Figure B1.6).

![Login](Login.png)

**Figure B1.6 Task to Actor Association Dashboard.**

Within the TasktoActorAssoc class, a function named MakeAssocation() was written. A number of programmatic issues had to be resolved in order to capture an unknown and unlimited amount of data posted from the student dashboard. In normal circumstances, variable data coming from a ‘web form’ is pre-determined including its key and value (key-value pairs). Allowing the student to send unlimited key-value pairs necessitated the production of a ‘while loop’ that populated an array. Within the array, an index contained a reference to the number sequence of the task as it appeared in the dashboard, along with the WebAppTask_ID reference (see Figure B1.7).
Once the WebAppTask_ID’s had been collected, the next part of the routine was to collect the four actors associated with the task. A subroutine was written to achieve this. It had been hoped to take advantage of PHP’s ability to create multidimensional arrays, for example, `array ( array("task1", “actor1_ID”, “actor2_ID”, “actor3_ID”, “actor4_ID”))` Implementing using this approach proved to be unreliable and an alternative solution was sought.

```php
// Collect The Tasks in an array.
// We will then know the quantity to be associated in 'e'
$task=array();
$e=1;
while ($_POST['task'.$e]) |
$task[$e]=$_POST['task'.$e];
$e++; |
```
Figure B1.8 Association Classes - Subroutine Logic.

Figure B1.8 represents a solution to the problems of displaying multiple associations in the student dashboard. This was perhaps the most challenging aspect of the development and one which took the most time to write, test and validate. Much of the understanding developed in writing and developing this subroutine was applied to other association classes, such as the `FunctionToActorAssoc()` association class and would also be used in the editing subroutines of these functions (see Appendix C1 - MakeAssociation PHP class). The subroutine provided the basis for the `FunctionToActorAssoc()`, with some modifications necessary to reflect variations in data labels. Iteration two proved to be stable in terms of its
ability to add associations. The next iteration focused on editing facilities within the student dashboard.

**Association Class – Third Iteration**

The second iteration of the association class provided a stable code base in which to explore further solutions to a number of programmatic problems. This was achieved in the third iteration of the prototype. Although the ‘add association’ function proved to be stable, an editing facility was also required to reflect the changing nature of requirements over the duration of the web project. It was found that only slight modifications were needed to this function and the advantages of using the webActor_ID as the foreign key was a central factor in its correct execution. The student ‘edit task to web actor association’ dashboard provided a list of all associations (see Figure B1.9 Edit Task to Web Actor Association).

**Editing - Task To Web Actors Association**

<table>
<thead>
<tr>
<th></th>
<th>Login</th>
<th>Register</th>
<th>View Products</th>
<th>Search Products</th>
<th>Shopping Basket</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Customer</td>
<td>Web Admin</td>
<td>Not Assigned</td>
<td>Not Assigned</td>
<td>Not Assigned</td>
</tr>
<tr>
<td>2</td>
<td>Customer</td>
<td>Web Admin</td>
<td>Shop Worker</td>
<td>Not Assigned</td>
<td>Not Assigned</td>
</tr>
<tr>
<td>3</td>
<td>Customer</td>
<td>Shop Worker</td>
<td>Not Assigned</td>
<td>Not Assigned</td>
<td>Not Assigned</td>
</tr>
<tr>
<td>4</td>
<td>Customer</td>
<td>Shop Worker</td>
<td>Not Assigned</td>
<td>Not Assigned</td>
<td>Not Assigned</td>
</tr>
<tr>
<td>5</td>
<td>Customer</td>
<td>Not Assigned</td>
<td>Not Assigned</td>
<td>Not Assigned</td>
<td>Not Assigned</td>
</tr>
</tbody>
</table>

**Figure B1.9 Edit Task to Web Actor Association.**

The `EditAssociation()` function within both the `FunctionToActorAssoc()` and `TaskToActorAssoc()`, classes collected ‘post data’ and initiated an ‘update’ query. This function utilised much of the logic developed in the `MakeAssociation()` function, with additional routines written to ensure data validity. Work continued on additional classes that enabled the student to add, edit and delete Web Actors; Tasks; Functional Requirements; and non-functional requirements.
Consistency, Completeness and Correctness and Dependencies Checking

Class – Fourth Iteration

One of the benefits of moving to a CAWE tool was an ability to impose constraints within a rules model. The rules model in turn provided a way forward to incorporate some of the ideas developed in the diagnosing and problem identification section of this section, especially relating to completeness of the SRS Document.

The rules model was represented in the `consistencyAndCompleteness()` function within the Dashboard class, where checks were to be performed to ensure that the student could not produce the SRS Document until certain rules within the model were met. Additionally, the class became host to a number of core functions that governed the student dashboard. For example, the functions it contained provided a visual representation of the meta-model, including those aspects which were completed or incomplete (see Figure B1.10).
Figure B1.10 CAWE Tool - Student Dashboard.

The student dashboard also controlled access to the editing, deletion and dependencies checking functions. For example, if an web actor was removed, a dependencies check was performed on the $\text{FunctionToActorAssoc()}$ and $\text{TaskToActorAssoc()}$ associations (see Figure B1.11). The student could then confirm deletion in acknowledgement of the dependencies checking routine.
Dependencies Check For Actor: Customer

Before Deleting, Please Check The Following:

A. The Following Tasks Are Associated With This Actor!
1: Login
2: Register
3: View Products
4: Search Products
5: Shopping Basket

B. The Following Functional Req's Are Associated With This Actor!
1: Shall: allow the user to login into the website from the home page.
2: Shall: allow the user to register

Do You Still Wish To Delete The Actor? Doing So Will Also Remove These Associations!

Figure B1.11 Dependencies Checking Function.

The consistencyAndCompleteness() function in the dashboard class also contained a routine that controlled the students ability to create the SRS document. Each time the dashboard was accessed the class was initiated. It then executed a number of functions to check for the completeness of the requirements. Dashboard objects were also generated that allowed the student to add, edit or delete data as appropriate. For example, the student may have edited the ‘statement of purpose’ or delete ‘tasks’. Once all of the steps were completed, the student dashboard allowed the student to view the SRS document via a highlighted button (see Figure B1.12).
The fourth iteration proved to be stable in terms of being able to add, edit, and delete user data and to check for the consistency, completeness and correctness of the requirements. Chapter 2 in Volume 1 highlighted the need capture and output the requirements in a ‘requirements specification document’ and suggested ways that this could be achieved. Stability of the web application and confidence in the validity of the user data, meant work could start on this aspect of the CAWE tool.
Requirements Specification Document Class – Fifth Iteration

The final iteration within the implementation phase of the CAWE tool included a mechanism to represent the user data within an overall SRS document. Since the consistencyAndCompleteness() function within the Dashboard class had already checked requirements, the resulting document would contain validated data and therefore deemed to be complete. In line with investigation into Requirements Specification Documentation, additional informational items were added, such as a version number and date stamps. Log data that was captured whenever a student performed an action within the CAWE tool provided the date and revision number that is displayed within the SRS document (see Figure B1.13).

Within the rest of the requirements specification, information entered by the student during the analysis stage was formatted and presented. Functional and non-functional requirements are also displayed. Associations are made explicit within the SRS document and each functional requirement has a unique id that was consistent even if the requirements are subsequently modified. Non-functional requirements were displayed at the end of the SRS document (see Figure B1.14, B1.15 and B1.16).
Figure B1.13 Requirements Specification Documentation – Version Control.

Project Description: As directed in the project brief, the website is to reflect the existing XYZ business objectives. Providing the user with their own personal portfolio, the site will be accessible from the intranet...

Statement Of Purpose: Create a prototype online Virtual Gallery for the School of Computing. This gallery will allow students to upload their work for view by friends, family, potential students and employers. It must therefore display the work in a usable and attractive way.

Objective1: test

Tasks And Associated Actors
Task Name: Login Actor: Student,
Task Name: Register Actor: Student,
Task Name: View Portfolio Actor: Student, Admin,
Task Name: Search Portfolio Actor: Student, Admin,

Functional Req’s And Associated Actors
Functional Req’s: Supports the ability for a student to upload content. Actors: Student,
Functional Req’s: Supports the ability for a student to register. Actors: Student,
Functional Req’s: Supports the ability for a student to login. Actors: Student,
Functional Req’s: Supports the ability for a student to edit content. Actors: Student,
The traceability of requirements was highlighted as being an issue in the second research cycle. Creating an identifier for each requirement was one
mechanism to allow the student to link the requirement through to the design documentation and into the implementation for validation purposes.

It was also important to format the requirements specification document so that it was usable by a range of people involved in the web project. Other group members, tutors and assessors would all need to read the information that it contained. The student would also need to print off the document for inclusion within the ICA submission. Two versions of the document were therefore offered to the student:

1. A screen only version that allowed the student to review the information quickly and easily.

2. A print version that allowed the student to print off information ready for inclusion within their ICA for assessment purposes. A printer friendly button was provided in the screen only version.

In order to facilitate traceability via the identifier system, during initial testing it was found that it was not possible to select a requirement and display it within the requirements specification document. On further investigation it was established that as requirements were constantly updated and deleted, the primary key of the WebAppFunction table was not a reliable mechanism to use for the identifier for each requirement. The MySQL documentation provided an explanation for this behaviour and a possible solution. It was found that unlike other relational database management systems, MySQL was able to ‘reuse primary keys’ once a record had been deleted, a behaviour that had not been envisaged in the initial design of the RequirementPattern class.

A more reliable approach was written and tested which would ensure forward and backwards traceability even if the requirements were modified. A RequirementPattern class was initiated from within the student dashboard once they had selected the ‘View Requirements Document’ button (see Figure B1.17 RequirementPattern Class).
Figure B1.17 RequirementPattern Class.

The class selected and transformed the data for insertion into the RequirementPattern table. A RequirementPattern_ID provided the unique identifier for each requirement, which in turn was used by the RequirementDocument class that formats and displayed the information to the student within the SRS document. A print version was accessible from within the screen, which in turn called an additional class named requirementDocumentationPrint that formatted the specification so that it was compatible for printers (see Appendix C2, requirementDocumentationPrint PHP class).

CAWE Tool Deployment

Prior to the website being launched, a number of additional classes had to be integrated within the CAWE tool to facilitate common tasks and to enhance the user experience. This included user registration; login; password recovery; password change; help and guidance; actor profiles; video tutorials; log data and student profile questionnaire.
To facilitate the student registration and to enable later tracking of individual students via the log system, the registration screen consumed a web service that provided a list of modules from the HEI student registry application. This would enable precise tracking of each action the student undertook within the CAWE tool and would be used for later analysis in the evaluation of the CAWE tool in use.

**Alpha Testing**

Stability of the registration system and user authentication was paramount, as it would become a mission critical tool in terms of use on a number of modules and in particular for the assessment process. Testing of individual classes which had taken place during the iterative prototyping stage had uncovered a number of issues and these had been resolved before moving to the next version. It was found that the validity of the data within the association classes was problematic and traced to an SQL query that was not correctly formed. Alpha testing was necessary to ensure that no conflicts or errors existed when executing functions and classes together. A number of dummy user accounts were created to ensure validity of data generated by the individual users. Alpha testing was conducted on a live server environment, but hidden behind a firewall to ensure no one else could access it.

During alpha testing a number of problems were identified:

1. Null data was being inserted into the Actor, Task and Website Objective tables. A knock on effect persisted within the association class, in particular PHP generated infinitive loops due to missing data. The application crashed at this point, resulting in a server time out problem. The issue was resolved by validating and cleaning POST data sent from the student dashboard and sending the student an appropriate message for them to address the missing form data.

2. MYSQL primary key column sequencing problem on the Actor and Task tables. A ‘referential integrity bug’ presented itself when deleting an Actor or Task where these had previously being associated. MYSQL reuses primary keys when using the ‘DELETE WHERE’ clause and this resulted in an ‘out of
sequence id pattern’ within the tables, which caused the application to crash. This is a documented feature of MYSQL which adversely affected testing until a solution could be found. The solution centred upon resequencing the primary key column using the ‘ALTER TABLE’ clause after deletion occurred in both Web Actor and Task tables. For example:

```
ALTER TABLE WebAppActor ORDER BY WebAppActor_ID ASC;
```

3. Due to the stateless nature of the platform and the object orientated approach for initiating functions within classes, it was necessary to adopt two technologies in order to ‘maintain state’. These included session variables, where key-value pairs reside on the server and can be accessed by the application as long as the session persists. Global variables were used in order to assign variables for use in multiple functions, as some classes were run simultaneously from within a single class call. Persistence of sessions presented the author with the majority of programmatic issues to resolve. An appropriate convention was setup to aid this, with global variables always declared in a group at the top of the relevant function.

4. A vulnerability was identified with user data, where an SQL query could accidentally delete data not belonging to the student. All SQL statements were re-written with the following additional clause: `WHERE Owner_ID = '$Owner_ID'`. Data that was being modified at this stage now had to belong to the student that had logged in. In addition, the vulnerability called into question the safety of holding all user data in one database. It was therefore decided to run a backup for the user data each day. An automatic database backup function was written for use in the final version, which relied on the UNIX ‘Cron’ call, a system that enables a time based job schedule to run at
predetermined intervals. It was later found that this feature was not accessible on the HEI server. Manual backups were therefore the only option available.

**Final Release**

A dedicated web address ‘www.scm.tees.ac.uk/WURF’ was set up to host the CAWE tool to ensure it could be easily accessed and that it was available outside of the institution’s firewall. To assist with future error tracing and to ensure integrity of student-generated data, two versions of the application were created to run concurrently:

1. A live application that would be used by staff and students.

2. A test application that would be used by the author to recreate and fix errors as they arose on the live site. Test data from the live site could be ported over to the test application in order to accurately recreate the error. Critical updates to the test application could be made before being applied to the live site.

Some differences were noted on the server running the live site. Most notably the version of PHP on the live server was 4.6 and on the test server it was PHP 5.3. The reason for this was legacy software running on the intranet. No major issues presented themselves due to the difference in PHP versions, but it did prevent the running of generic PHP classes that had been developed for automatic database backup, as this used PHP 5 libraries to facilitate file output.
Appendix B1.1 - Software Requirements Specification

xxxxx Access Requirements

Status: Second Release

Author: xxxxx

Date of issue: xxxxx

Reference: xxxxx

Number of pages: xxx

Table of Contents

1 DOCUMENT CONTROL 371
2 EXECUTIVE SUMMARY 371
   2.1 MISSION STATEMENT FOR THE PROJECT 371
3 PRIORITY OF REQUIREMENTS 371
4 POTENTIAL IMPLEMENTATION SOLUTIONS 372
4. HOW AND WHERE WILL USERS BE ABLE TO ACCESS? 373
   4.1 USER ACCESS CONTROLS 373
      4.1.1 Must Have ................................................................. 373
      4.1.2 Should Have .............................................................. 373
      4.1.3 Could Have ............................................................... 373
      4.1.4 Would Be Nice To Have ............................................. 373
      4.1.5 Rejected ................................................................. 373
Executive Summary

Mission Statement for the Project

The mission of this project is to “put in place systems that enable The xxxxxx to become the point of first resort for anyone who wants to access a comprehensive archive of material from the United Kingdom Web domain.”\(^1\)

The Access Requirements project will generate a list of requirements for accessing information stored within the xxxxxxx. The project has a fixed time scope of xxx weeks, and is intended to gather the majority of the requirements, including all the key requirements and possibly a list of areas that require further investigation.

The scope of this project is limited to the Access to the xxxxxx, i.e. harvest / ingest of websites, other archive material (e.g. books, journals and newspapers) are out of scope.

Priority of Requirements

Once the requirements had been identified (see Error! Reference source not found. for the methodology used to identify requirements), they were categorised into one of five priorities, i.e.:

- “Must Have” requirements are basic functionality that is essential for the web archive;
• “Should Have” requirements are not essential for an initial version of the Web Archive, however they will have a substantial influence on the impact of the Web Archive, its usability and presentation and should be implemented as soon as possible;

• “Could Have” requirements are not essential for the Web Archive, however they are useful functionality that could be implemented in future versions of the Web Archive.

• “Would Be Nice To Have” requirements are not essential for the Web Archive; however they may be useful for a later version of the Web Archive, although they have a low priority.

• “Rejected” requirements are those which are either out of scope of this project, undesirable or untenable / undeliverable.

The assignment of priority to requirements has been done through direct ranking by external sources, e.g. the Oxford University Library Service survey, and from consensus of opinion from internal workshops.

**Potential Implementation Solutions**

During the course of gathering requirements, there were numerous times where potential implementation solutions have been suggested. These potential implementation solutions are out of scope of this document. However, they have been captured and for completeness included with the underlying requirement. They can be seen as indented italic text below the underlying requirement.
4. How and where will users be able to access?

There are three fundamental requirement areas to consider here, i.e. User Access Controls, Access Routes and Material Access Control.

User Access Controls, which are requirements focused on users, i.e. how to define the level of access for different types of user.

Access Routes, which are requirements focused on where a user can gain access to archived material.

Material Access Controls, which are requirements focused on archived material, i.e. how to define the level of availability for different material.

**User Access Controls**

*Must Have*

1. A level of access that does not require Username and Password, i.e. for unregistered users.

*Should Have*

2. Functionality to allow different access rights for different types of user, i.e. multiple levels of access (e.g. Reader, Registered Researcher and Non-Registered Researcher).

   *If there is a requirement (from Legal Deposit legislation) to restrict access, then it would be better to have this restriction controlled using a login, which will allow access through the internet anywhere, i.e. not just in reading rooms.*

3. The ability to offer value added services linked to a user, e.g. user created collections, and favourites.

   *This could be facilitated through the use of a user account or an identification layer, e.g. user registration, identification and password or Virtual Readers Card. Any user registration process for this should be user friendly, i.e. not onerous.*

*Could Have*

4. Remote access for site owners to the archived version of their site, even if it is not publicly available.

*Would Be Nice To Have*

5. Access controlled through an existing standard, e.g. Shibboleth.

*No User Access Control requirements have been given this priority.*

*Rejected*

6. Must be a way to configure user access. The minimum requirement will be to have access set by generic user profile and the maximum requirement will be to have access set on a user by user basis. This requirement has been rejected as there have been repeated suggestions that restrictions should not be levied at individuals.
Appendix B1.2 - Student Email for Survey

Dear Student.

Following on from last weeks’ lecture announcement, please find link to the e-WURF Evaluation Questionnaire for Online Business Systems, as required for your ICA!

Your views are greatly appreciated and should only take 5mins to fill in. The data that you submit will not be used for any other purpose.

http://www.scm.tees.ac.uk/WURF/questionnaire2.php?userID=f6047269

Please cut and paste into your browsers address bar if the above link is not active

All the best.

Andrew Bingham
Senior Lecturer (Web Development)
Associate Teaching Fellow
School Of Computing
University Of Teesside
TS1 3BA

T: 01642 342666
E: a.p.bingham@tees.ac.uk
Appendix C1 – MakeAssociation Class

```php
<?php
session_start();

//////////////////////////////////////////////////
// MakeAssociation Function - Actors And Tasks Association
//////////////////////////////////////////////////
include ('../common.php');
DBConnect();

//COLLECT GET data and create an array
$formValueGET=array();
foreach ($_GET as $key => $value) {
    $formValueGET[$key] = strip_tags($value);
}

//Conditional Statement To Check If We Are Adding or Editing Associations
$edit = $formValueGET['edit'];
$fromAdmin = $_GET['fromAdmin'];

if($edit == "true"){
    EditAssociation($Host, $User, $Password, $DBName, $table_1, $table_2, $table_3, $table_4, $table_5, $table_6, $table_7, $table_8);
} else {
    MakeAssociation($Host, $User, $Password, $DBName, $table_1, $table_2, $table_3, $table_4, $table_5, $table_6, $table_7, $table_8, $fromAdmin);
}

```
$Link = mysql_connect($Host, $User, $Password);
$Owner_ID = $_SESSION['Owner_ID'];

// We need to collect the task ID and the next 4 actor ID's.
$task = array();
$e = 1;
while($_POST['task'.$e]){  
    $task[$e] = $_POST['task'.$e];
    //echo $image[$i].'  
    $e++;
}
$e = $e -1;

// var $b will count the total tasks submitted by the user
$b = 1;

// var $s as our comparison operator
$s = 1;

// Collect the task value from the array using while loop
while($s<=$e){

// For collecting multi-dimensional array values
    // var $q = the actor array
    // var $k = actor / task association array
    $q = 1;
    $k = 1;
// A maximum of 4 actors can be associated with each task
// Therefore, run the nested while loop 4 times to see if we have an association with the task that has been submitted by the user
    while($c <= 4){

```
$actor[$q] = $_POST['actorAssoc'.'_'.$k.'_'.$b];

// For each task / actor association, create a new record
$Query = "INSERT INTO $table_7 VALUES ('0','".$task[$b]."','".$actor[$q]."','$Owner_ID')";

mysql_db_query ($DBName, $Query, $Link);
$c++;
$k++;
$q++;
}

// Close nested while loop

$s++;
$b++;
}

// Close while loop

// Invoke Subsystem Usage Data
global $Subsystem;
$Subsystem = "add task to actor association";
include "../classes/class.Log.php";

// End Usage Data

// Determine if function has been called from student dashboard or from the initial setup screen
if($fromAdmin==true){
    // Return to student Dashboard
$message = "Tasks and Actors Associated Successfully!"

header("Location: ../index.php?message=$message");

}else{
    //Move on one step in the setup process
    $message = "Tasks and Actors Associated Successfully!"

    header("Location: ../Functions.php?message=$message");
}

} //Close Function


////////////////////////////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////////////////////////////

// EditAssociation - Actors And Tasks Link

//on WURF_WebAppTaskAssoc

////////////////////////////////////////////////////////////////////////////////////

function EditAssociation($Host, $User, $Password, $DBName, $table_1, $table_2, $table_3, $table_4, $table_5, $table_6, $table_7, $table_8)
{
    $Link = mysql_connect($Host, $User, $Password);
    $Owner_ID = $_SESSION['Owner_ID'];

    //We Need To Collect The Task ID And the next 4 Actor ID's.
    $task=array();
    $e=1;
    while($_POST['task'.$e]){
        $task[$e]=$_POST['task'.$e];
    }
//echo $image[$i].’ ’;
$e++;
}

//We Are Going To Have build this function differently,

//But we DO have a known quantity for the Tasks! //Lets collect the Tasks using the SELECT query in EditAssociate_WebAppTaskTo_Actors.php

//Then do a while loop 4 times to collect each actor!

//Get Our Task List Again so that we know what we are expecting and can use result set as a comparison

$Query = "SELECT * FROM $table_5 WHERE Owner_ID = '$Owner_ID' ORDER BY 'WebAppTask_ID' ASC";

$Result = mysql_db_query($DBName, $Query, $Link);
while ($Row = mysql_fetch_array($Result)){
  $taskId = $Row['WebAppTask_ID'];
  $taskName = $Row['taskName'];
}

//Close while loop

//We Need To Collect The Task ID And the next 4 Actor ID's that have been submitted by the user
$t=asarray();
$e=1;
while($_POST['task'.$e]){  
  $task[$e]=$_POST['task'.$e];
  echo $image[$i].’ ’;
  $e++;
}  
} //close while loop
//var $b will count the tasks submitted by the user
    $b = 1;

//var $s will be our comparison operator
    $s = 1;

//Collect the task from the array
    while($s<=$e){

    //For collecting our multi-dimensional array values
        $q = 1;
        $c = 1;
        $k = 1;
        $once = true ;

//We have a fixed ability for 4 actors to be associated with each task
//Run the loop 4 times to see if we have an association with the task that has been submitted
    while($c <= 4){

    //SELECT ON $task[$b] to find the TaskToActorAssoc_ID // But We will need to a switch to say record + 1;

    if($once==true){

        $Query = "SELECT TaskToActorAssoc_ID FROM $table_7 WHERE WebAppTask_ID = "'".$task[$b]."'" && Owner_ID = "$Owner_ID" ORDER BY 'WebAppTask_ID' ASC";
    }
}
} else {

    // Then select the next 3 TaskToActorAssoc_ID in sequence

    $ID = $TaskToActorAssoc_ID + 1;

    $Query = "SELECT TaskToActorAssoc_ID FROM $table_7 WHERE TaskToActorAssoc_ID = "'.$.ID.'";"

} // close conditional statement

$Result = mysql_db_query($DBName, $Query, $Link);

$Row = mysql_fetch_array($Result);

$TaskToActorAssoc_ID = $Row['TaskToActorAssoc_ID'];

$actor[$q] = $_POST['actorAssoc'.'_'.k.'_'.b];

$once = false;

// UPDATE THE RECORD

$Query2 = "UPDATE $table_7 SET WebAppTask_ID = "'.$.task[$b].'", WebAppActor_ID = "'.$.actor[$q].'" WHERE TaskToActorAssoc_ID = "'.$.TaskToActorAssoc_ID.'";"

mysql_db_query ($DBName, $Query2, $Link);
//Invoke Subsystem Usage Data

global $Subsystem;
$Subsystem = "edit task to actor association";
include ".../classes/class.Log.php";

//End Usage Data
$message = "Tasks and Actors Association Updated Successfully!";
header("Location: ../index.php?message=$message");

}//Close Function

?>
Appendix C2 – RequirementDocumentationPrint Class

```php
<?php
session_start();

//class.requirementDocumentationPrint.php

/////////////////////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////////////////////
include ('../common.php');
DBConnect();

GenerateDocumentation($Host, $User, $Password, $DBName, $table_1, $table_2, $table_3, $table_4, $table_5, $table_6, $table_6_A, $table_7, $table_8, $table_9, $table_10, $table_11, $table_12, $table_13, $table_14, $table_14_A, $table_15, $table_16, $table_17);

/////////////////////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////////////////////

//Development Notes 12/08/09

// This Function Generates The Requirement Documentation - Printable Version******

//This Class is called by the following class:
class.requirementPatterns.php

//Initial Prototype to Produce HTML

//This can be converted into a PDF file later if required, once PDF extension is installed on server.

/////////////////////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////////////////////
```
function GenerateDocumentation($Host, $User, $Password, $DBName, $table_1, $table_2, $table_3, $table_4, $table_5, $table_6, $table_6_A, $table_7, $table_8, $table_9, $table_10, $table_11, $table_12, $table_13, $table_14, $table_14_A, $table_15, $table_16, $table_17 ){

    //Object Model For The Document

    // Rewind array's internal pointer to the first element and returns
    // the value of the first array element.
    function Array_Dimensional_Reset(&$arrRef) {
        foreach ($arrRef as $key=>$val) {
            if (is_array($val)) {
                $this->Array_Dimensional_Reset($val);
                reset($arrRef[$key]);
            }
        }
    }

    //include "checkLoginSession.php";
    $Link = mysql_connect($Host, $User, $Password);
    $Owner_ID  = $_SESSION['Owner_ID'] ;

    //1. Project Details - First lets select the project details

    $Query = "SELECT t1.Owner_ID, t1.title, t1.description, t2.firstN,
    t2.lastN, t2.email, t2.Owner_ID FROM $table_16 AS t1, $table_1 AS
t2 WHERE t2.Owner_ID = 'Owner_ID' AND t1.Owner_ID = 'Owner_ID';

$Result = mysql_db_query($DBName, $Query, $Link);

$Row = mysql_fetch_array($Result);

$title = $Row['title'] ;

$description = $Row['description'] ;

$email = $Row['email'] ;

$Owner_firstName = $Row['firstN'] ;

$Owner_lastName = $Row['lastName'] ;

//Use HEREDOC method to build the first object of our document
$Doc_P1 = <<<DOC
<hr />
<div>Project Owner: <span class="bluetext">$Owner_firstName $Owner_lastName </span></div><br />
<div>Owner Email Address: <span class="bluetext">$email@tees.ac.uk </span></div><br />

<div>Project Description: <span class="bluetext">$description</span></div>

//Close HEREDOC
DOC;
//2. Website Objectives

$Query = "SELECT * FROM $table_4 WHERE Owner_ID = '$Owner_ID'";
$result = mysql_db_query($DBName, $Query, $Link);
$i = 0;
$k = 1;
while ($Row = mysql_fetch_array($Result)){
    $obj.$i = $Row['ObjName'];
    $objOut = $objOut. "Objective".$k.":<div>Objective".$k.":<span class='bluetext'>$obj.$i</span></div>";
    $i++;
    $k++;
} //close while loop

//3. Statement Of Purpose

$Query2 = "SELECT * FROM $table_3 WHERE Owner_ID = '$Owner_ID'";
$result2 = mysql_db_query($DBName, $Query2, $Link);
$Row2 = mysql_fetch_array($Result2);

$statementTXT = $Row2['statementTXT'];

$Doc_P2 = "<hr />
<div>Statement Of Purpose AND Website Objectives:</div><br />
<div>Statement Of Purpose: <span class='bluetext'>$statementTXT</span></div>";

$objOut
<hr />
//Close HEREDOC
DOC1;

//4. Task And Associated Actors For Project
//This is the tricky bit!!!!!

$Query3 = "SELECT WebAppTask_ID, WebAppActor_ID, Owner_ID FROM $table_7 WHERE Owner_ID = '$Owner_ID'";

$Result3 = mysql_db_query($DBName, $Query3, $Link);
$i=1;
$Doc_P3 = "<br /><div>Tasks And Associated Actors</div>";
$z = 1;

//Get The Task ID's
//LOOP A
while ($Row3 = mysql_fetch_array($Result3)){
    $TaskName[] = $Row3['WebAppTask_ID'];
    $myActor[] = $Row3['WebAppActor_ID'];
    $z++;
} //CLOSE LOOP A

//We Now Have Our Comparison Data Which We Can Use In The Loop To Look For The Actors Associated With Each Task
//Remove duplicate values from the array, so we only print off consistent requirements, not duplicates.
$result = array_unique($TaskName);
$j=1;
$a=1;

//LOOP B
for ($i=0; $i<=$z; $i++){
    //Select The WebActors Table
    if(empty($result[$i])){
        //null value returned, ensure we don’t see this in our document
    }else{
        //Now We Look Up The 4 Actors Associated For This Task
        $Query3_3 = "SELECT WebAppActor_ID, WebAppTask_ID FROM $table_7 WHERE WebAppTask_ID = '$result[$i]'";
        $Result3_3 = mysql_db_query($DBName, $Query3_3, $Link);
        $h = 1;

        //LOOP C
        while($Row3_3 = mysql_fetch_array($Result3_3)){
            $ActorKey = $Row3_3['WebAppActor_ID'];
            $TaskKey = $Row3_3['WebAppTask_ID'];

            //***Additional Query To Get The Task Name (2/9/09)
            $Query3_3_4 = "SELECT WebAppTask_ID, taskName FROM $table_5 WHERE WebAppTask_ID = '$TaskKey'";
            $Result3_3_4 = mysql_db_query($DBName, $Query3_3_4, $Link);
            $Row3_3_4 = mysql_fetch_array($Result3_3_4);
            $Taskname = $Row3_3_4['taskName'];

            //Now We Can Do A SELECT To Return Actor Data
            $Query3_3_3 = "SELECT WebAppActor_ID, actorName FROM $table_6 WHERE WebAppActor_ID = '$ActorKey'";
            $Result3_3_3 = mysql_db_query($DBName, $Query3_3_3, $Link);
$Row3_3_3 = mysql_fetch_array($Result3_3_3);
if(empty($Row3_3_3['actorName'])){
    //Null record returned
}else{
    $ActorName_Task.$a = $Row3_3_3['actorName'];
    $myActorList_Task = $myActorList_Task."".]+$ActorName_Task.$a"","";

    $h ++;
    $a++;

};//Close if empty conditional statement
};//CLOSE LOOP C

$a =1;
$Doc_P3 = $Doc_P3.""Task Name: <span class='bluetext'>$Taskname</span>"".>&nbsp;&nbsp;Actor: <span class='bluetext'>$myActorList_Task</span>""<br />

$myActorList_Task = "";
};//Close if empty conditional statement
$j++;
};//CLOSE LOOP B

$g=1;
$s =1;
$i++;

//****END OF TASK & ACTORS ASSOCIATION*****

//5. Functions And Actor Associations.
$Query3 = "SELECT WebAppFunction_ID, WebAppActor_ID, Owner_ID FROM $table_10 WHERE Owner_ID = '$Owner_ID';"

$Result3 = mysql_db_query($DBName, $Query3, $Link);
$i=1;

$Doc_P4 = "<br /><div>Functional Req's And Associated Actors</div>";
$z = 1 ;

//Get The Task ID's
//LOOP A
while ($Row3 = mysql_fetch_array($Result3)){
$FunctionName[] = $Row3['WebAppFunction_ID'];
$myActor[] = $Row3['WebAppActor_ID'];
$z++;
}//CLOSE LOOP A

//We Now Have Our Unique Number Which We Can Use In The Loop To Look For The Actors Associated With The Function
$result = array_unique($FunctionName);
$j=1;
//LOOP B
for ($i=0;$i<=$z; $i++){ //Select The WebActors Table if(empty($result[$i])){
//Null data returned, ensure this is not displayed in the documentation
}else{
//Now We Look Up The Actors For Just That Task

$Query3_3 = "SELECT WebAppActor_ID, WebAppFunction_ID FROM $table_10 WHERE WebAppFunction_ID = '$result[$i]';";

$Result3_3 = mysql_db_query($DBName, $Query3_3, $Link);
$h = 1;

//LOOP C
while($Row3_3 = mysql_fetch_array($Result3_3)){

$ActorKey = $Row3_3['WebAppActor_ID'];
$FunctionKey = $Row3_3['WebAppFunction_ID'];

///***Additional Query To Get The Function Name (2/9/09)
$Query3_3_4 = "SELECT WebAppFunction_ID, functionText FROM $table_9 WHERE WebAppFunction_ID = '$FunctionKey';";
$Result3_3_4 = mysql_db_query($DBName, $Query3_3_4, $Link);
$Row3_3_4 = mysql_fetch_array($Result3_3_4);

$Functionname = $Row3_3_4['functionText'];

//Now We Can Do SELECT To LOOK UP Who The Actor Is On The Actors Table
$Query3_3_3 = "SELECT WebAppActor_ID, actorName FROM $table_6 WHERE WebAppActor_ID = '$ActorKey';";

$Result3_3_3 = mysql_db_query($DBName, $Query3_3_3, $Link);
$Row3_3_3 = mysql_fetch_array($Result3_3_3);
if(empty($Row3_3_3['actorName'])){    
    //Null data returned
}
}'else{
$ActorName_Function.$a = $Row3_3_3['actorName'];

$myActorList_Function = $myActorList_Function.".$ActorName_Function.$a"," ;
$h ++;
$a ++;

}//Close If EMPTY

};//CLOSE LOOP C
$a =1;

$myActorList_Function = "";
}//CLOSE IF EMPTY
$j++;
}//CLOSE LOOP B

$g=1;
$s =1;
$i++;

//****END OF TASK & ACTORS ASSOCIATION******/

/* This fix is to align the id in the correct order in the RequirementPattern table to ensure functional requirement id’s are in the correct order within the requirements document*/

$Query_alter = "ALTER TABLE $table_17 ORDER BY RequirementPattern_ID ASC";
mysql_db_query($DBName, $Query_alter, $Link);
Now We Generate Each Requirement Pattern

$Query5 = "SELECT * FROM $table_17 WHERE Owner_ID = '$Owner_ID'
ORDER BY RequirementPattern_ID ASC";

$Result5 = mysql_db_query($DBName, $Query5, $Link);

$Doc_P5 = '<div class='breakhere'></div>'

Requirements Specification
Document</div><br />
Key: FR-x = Functional Requirement || NFR-x = Non-Functional Requirement</div>"

$p = 1 ;
while ($Row5 = mysql_fetch_array($Result5)){
    $FunctionnameQ = $Row5['name'] ;
    $Doc_P5 = $Doc_P5. "<div class='PatContainer'>";
    $Doc_P5 = $Doc_P5. "$Row5[ReqRef]";
    //We can tailor the description feild here
    $type = $Row5['type'] ;

    if($type=="Non-Functional Requirement"){
        $name = $Row5['name'];
        $description = $Row5['description'];
    }else{
        $name = "&nbsp;&nbsp;&nbsp;Functional Requirement $p";
        $description = $FunctionnameQ ;
    }
}
$Doc_P5 = $Doc_P5. "<div class='Row1Cent'>$name</div>";

$Doc_P5 = $Doc_P5. "<div class='Row1Right'>$Row5[type]</div>";

$Doc_P5 = $Doc_P5. "<div class='Row2'>Description: <span class='bluetext'> $description</span></div>";

//*****Here we need to do a Sub Query to get the Task Name for the Function ID.****
//*****Here we need to do a Sub Query to get the Actor Name .****
$Actor1 = $Row5['Actor1_ID'];
$Actor2 = $Row5['Actor2_ID'];
$Actor3 = $Row5['Actor3_ID'];
$Actor4 = $Row5['Actor4_ID'];
if((empty($Actor1)) && (empty($Actor2)) && (empty($Actor3)) && (empty($Actor4))){
    $actorEmpty = true ;
}
$a =1 ;
$s = 1;
for ($f=0;$f<=3; $f++){
    if($s==1){
        $q = "WebAppActor_ID = '$Actor1'";
    }else if($s==2){
        $q = "WebAppActor_ID = '$Actor2'";
    }else if($s==3){
        $q = "WebAppActor_ID = '$Actor3'";
    }else{
$q = "WebAppActor_ID = '$Actor4'";
}

// Let's Look Up Who The Actor is
$Query5_2 = "SELECT WebAppActor_ID, actorName FROM $table_6 WHERE ".$q."";
$Result5_2 = mysql_db_query($DBName, $Query5_2, $Link);
$Row5_2 = mysql_fetch_array($Result5_2);
$ActorName.$a = $Row5_2['actorName'];
$WebAppActor_ID = $Row5_2['WebAppActor_ID'];

// Check that there is a profile for the actor
$Query5_3 = "SELECT WebAppActor_ID, actorName FROM $table_6_A WHERE ".$q."";
$Result5_3 = mysql_db_query($DBName, $Query5_3, $Link);
$Row5_3 = mysql_fetch_array($Result5_3);
$check = $Row5_3['WebAppActor_ID'];
if(empty($check)){
$myActorList = " ".$myActorList.$ActorName.$a."", " ;
} else{
$myActorList = " ".$myActorList." ".<a href='../ViewActorProfile.php?WebAppActor_ID=$WebAppActor_ID' target='_blank'>".$ActorName.$a."</a>"."", " ;
}

// Let's see if we can have a statement if no actors are assigned
if($actorEmpty== true) {
$myActorList = "No Actors Assigned!";
}
$s = $s + 1;
$a++;
$a = 1;
$s = 1;
$type = $Row5['type'];
if($type == "Non-Functional Requirement"){

$myActorList = "Not Applicable To Non-functional Requirements";
$addinfo = "";
$comments = "Additional Req's: <span class=bluetext>".$Row5['comments']."</span>";
}else{
$comments = "Comments:<span class=bluetext>".$Row5['comments']."</span>";
$addinfo = "<span class=bluetextsmall> <br />These Actors will need to be taken into consideration when designing/developing this requirement!</span>";
}

$Doc_P5 = $Doc_P5. "<div class='Row4'>Associated Actors: <span class=bluetext>$myActorList $addinfo</span></div>

//Now the comments from the templatePattern table
$Doc_P5 = $Doc_P5. "<div class='Row5'> $comments</div>
$Doc_P5 = $Doc_P5. "</div><div class='breakhere'></div>
$p++;
$myActorList = "";
}
//T2;

//Invoke Log and Usage data
global $Subsystem;
$Subsystem = "create requirements document print version";
include "./classes/class.Log.php";
//End Invoke Log
};//Close GeneratePatterns Function
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<link rel="stylesheet" type="text/css" href="../styles1.css"/>
<style type="text/css">
/*Requirement Pattern CSS */
.PatContainer{
position:relative;
}
height: 300px;
width: 600px;
/*background-color:#69F;*/

border-left-width:medium;
border-left-color:#333;
border-left-style:solid;

border-top-width:medium;
border-top-color:#333;
border-top-style:solid;

border-bottom-width:medium;
border-bottom-color:#333;
border-bottom-style:solid;

border-right-width:medium;
border-right-color:#333;
border-right-style:solid;
margin-top: 5px;
}

.Row1Left{
position:absolute;
left: 0px;
top:0px;
height: 50px;
Appendix C3 – Student Opinion Survey

OBS Module Responses:

Q1. Have You Used e-WURF for the Online Business Systems Module?

Yes: 100%
No: 0%

Q2. Have you been able to produce a requirements specification document?

Yes: 93%
No: 7%

Q3. Do you feel that the web development process has been enhanced by using e-WURF?

Yes: 96%
No: 4%

Q4. Did you use the e-WURF assistant?

Yes
No
Yes: 85%
No: 15%

Q5. Please rate the following in terms of usefulness in the e-wurf process

Statement Of Purpose

<table>
<thead>
<tr>
<th>Rating</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4%</td>
</tr>
<tr>
<td>2</td>
<td>4%</td>
</tr>
<tr>
<td>3</td>
<td>22%</td>
</tr>
<tr>
<td>4</td>
<td>52%</td>
</tr>
<tr>
<td>5</td>
<td>-19%</td>
</tr>
</tbody>
</table>

Objectives

<table>
<thead>
<tr>
<th>Rating</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4%</td>
</tr>
<tr>
<td>2</td>
<td>0%</td>
</tr>
<tr>
<td>3</td>
<td>11%</td>
</tr>
<tr>
<td>4</td>
<td>41%</td>
</tr>
<tr>
<td>5</td>
<td>-44%</td>
</tr>
</tbody>
</table>

Tasks

<table>
<thead>
<tr>
<th>Rating</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0%</td>
</tr>
<tr>
<td>2</td>
<td>0%</td>
</tr>
<tr>
<td>3</td>
<td>22%</td>
</tr>
<tr>
<td>4</td>
<td>59%</td>
</tr>
<tr>
<td>5</td>
<td>-19%</td>
</tr>
</tbody>
</table>
Actors and Actor Profiles

1 - least useful - 0%
2   11%
3   15%
4   48%
5 - most useful - 26%

Task To Actors Association

1 - least useful - 0%
2   7%
3   22%
4   52%
5 - most useful - 19%

Functional Requirements

1 - least useful - 0%
2   4%
3   22%
4   52%
5 - most useful - 22%
Functional Req to Actor Association

1 - least useful - 4%
2 - 4%
3 - 35%
4 - 42%
5 - most useful -15%

User Interface (non-functional Req)

1 - least useful - 0%
2 - 4%
3 - 46%
4 - 42%
5 - most useful -8%

Marketing (non-functional Req)

1 - least useful - 0%
2 - 15%
3 - 38%
4 - 38%
5 - most useful -8%
Usability (non-functional Req)

1 - least useful - 0%
2   12%
3   27%
4   42%
5 - most useful -19%

Accessibility (non-functional Req)

1 - least useful - 0%
2   8%
3   15%
4   58%
5 - most useful -19%

Technical (non-functional Req)

1 - least useful - 0%
2   4%
3   33%
4   48%
5 - most useful -15%
Did you understand the difference between functional and non-functional requirements after you had used eWURF

Yes: 92%
No: 8%
Q6. Did you encounter any difficulties with the following

Statement Of Purpose

Yes: 0%
No: 100%

Objectives

Yes: 4%
No: 96%

Tasks

Yes: 8%
No: 92%
Actors and Actor Profiles

Yes: 4%
No: 96%

Task To Actors Association

Yes: 8%
No: 92%

Functional Requirements

Yes: 12%
No: 88%
Functional Req to Actor Association

Yes: 15%
No: 85%

User Interface (non-functional Req)

Yes: 4%
No: 96%

Marketing (non-functional Req)

Yes: 4%
No: 96%
Usability (non-functional Req)

Yes: 0%
No: 100%

Accessibility (non-functional Req)

Yes: 4%
No: 96%

Technical (non-functional Req)

Yes: 8%
No: 92%
e-WURF Assistant

Yes: 4%
No: 96%

Requirements Specification Document

Yes: 8%
No: 92%

If you answered yes to any of the above, did you contact e-WURF about this

Yes: 24%
No: 76%
If you answered yes to the above, was the problem resolved?

Yes: 25%
No: 75%

If you did not use e-WURF, have you used another requirements tool?

Yes: 6%
No: 94%

Are there any additions/modifications that you would like to see? Please enter them in the box below (Student Responses unedited).

- if you enter data into one of the sections and exit by mistake it should save it and say its unfinished, since i clicked the banner a number of times by mistake.

save link as .doc, so can print document in word. heard of difficulties printing from other students.

- To close the help dialog box you have to scroll back to the top to select the close link. A additional close link at the bottom would help prevent this.

- Better english skills being utilised on the site, example "build" being used where “Built” should have been.

- Consider the diverse browser (compatibility) or the platforms the website will run on as well as the requirements for mobile applications.

- When using the e-WURF Assistant you can only close the popup box by scrolling back to the top of the page. A bookmark back to the top of the page or close function in the footer would prevent additional scrolling.

- The whole structure felt a bit too rigid, giving no flexibility for any of the requirements.
- The ability to group 'Actors' into primary and secondary (even tertiary) target audience groups. This would make it easy to identify the most important actors.
- Add a PDF Format to see the document on the screen without necessary have an internet connection.
- No this is ok
- Consider carefully where radio buttons and where check-boxes are appropriate.

- Clarification on how to remove tasks. Formatting on associated tasks another areas could be improved, this maybe a Safari bug, a lot of scrolling is required. Sometimes radio buttons are not on the same line as the related answer, this can be seen in the Actor Profile page. I also nearly missed out creating Actor Profiles as they did not show under the Incomplete tasks on the home page.

- Although I managed to complete the accessibility and usability non-functional requirements sections I found that it could have been made clearer as to what information was trying to be received.
WAU Module Responses:

Q1. Have You Used e-WURF for the Web Authoring Module?

Yes: 86%
No: 14%

Q2. Have you been able to produce a requirements specification document?

Yes: 92%
No: 8%

Q3. Do you feel that the web development process has been enhanced by using e-WURF?

Yes: 92%
No: 8%
Q4. Did you use the e-WURF assistant?

Yes: 83%
No: 17%

Q5. Please rate the following in terms of usefulness in the e-wurf process

Statement Of Purpose

1 - least useful - 0%
2 - 0%
3 - 25%
4 - 50%
5 - most useful - 25%

Objectives

1 - least useful - 0%
2 - 0%
3 - 0%
4 - 42%
5 - most useful - 58%
Tasks

1 - least useful - 0%
2          0%
3          0%
4          42%
5 - most useful -58%

Actors and Actor Profiles

1 - least useful - 0%
2          0%
3          8%
4          58%
5 - most useful -33%

Task To Actors Association

1 - least useful - 0%
2          0%
3          8%
4          58%
5 - most useful -33%
Functional Requirements

1 - least useful - 0%
2          8%
3          8%
4          42%
5 - most useful - 42%

Functional Req to Actor Association

1 - least useful - 0%
2          8%
3          8%
4          42%
5 - most useful - 42%

User Interface (non-functional Req)

1 - least useful - 0%
2          8%
3          17%
4          33%
5 - most useful - 42%
Marketing (non-functional Req)

1 - least useful - 8%
2 0%
3 8%
4 42%
5 - most useful -42%

Usability (non-functional Req)

1 - least useful - 8%
2 0%
3 17%
4 33%
5 - most useful -42%

Accessibility (non-functional Req)

1 - least useful - 8%
2 0%
3 25%
4 25%
5 - most useful -42%
Technical (non-functional Req)

1 - least useful - 8%
2 - 0%
3 - 8%
4 - 42%
5 - most useful - 42%

e-WURF Assistant

1 - least useful - 0%
2 - 0%
3 - 25%
4 - 17%
5 - most useful - 58%

Requirements Specification Document

1 - least useful - 0%
2 - 0%
3 - 0%
4 - 42%
5 - most useful - 58%
Did you understand the difference between functional and non-functional requirements after you had used eWURF

Yes: 83%
No: 17%

Q6. Did you encounter any difficulties with the following

Statement Of Purpose

Yes: 0%
No: 100%

Objectives

Yes: 8%
No: 92%
Tasks

Yes: 8%
No: 92%

Actors and Actor Profiles

Yes: 0%
No: 100%

Task To Actors Association

Yes: 0%
No: 100%
Functional Requirements

Yes: 0%
No: 100%

Functional Req to Actor Association

Yes: 0%
No: 100%

User Interface (non-functional Req)

Yes: 0%
No: 100%
Marketing (non-functional Req)

Yes: 0%
No: 100%

Usability (non-functional Req)

Yes: 0%
No: 100%

Accessibility (non-functional Req)

Yes: 0%
No: 100%
Technical (non-functional Req)

Yes: 0%
No: 100%

e-WURF Assistant

Yes: 0%
No: 100%

Requirements Specification Document

Yes: 8%
No: 92%
If you answered yes to any of the above, did you contact e-WURF about this?

Yes: 14%
No: 86%

If you answered yes to the above, was the problem resolved?

Yes: 33%
No: 67%

If you did not use e-WURF, have you used another requirements tool?

Yes: 13%
No: 88%

Are there any additions/modifications that you would like to see? Please enter them in the box below (Student Responses unedited).

No, Very well developed.

When attempting to use the print safe option upon completion i found that if alot of information had been entered then the forms wouldnt accomodate it and so would not be displayed. Other than that it was a very useful assistance tool.
Appendix C4 – Usage Data Tables

Module: OBS - Assessment Mark vs Usage Relationship Analysis

HA1: Increased usage of the CAWE tool will result in higher marks for requirements analysis.

HA0: Increased usage of the CAWE tool will not result in higher marks for requirements analysis.

### Test 1. Requirements documents produced vs Marks

<table>
<thead>
<tr>
<th>Measure</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spearman's Rank Correlation Coefficient r</td>
<td>0.226230711</td>
</tr>
<tr>
<td>Statistical Significance of Correlation Two-Tailed T Test</td>
<td>0.325</td>
</tr>
</tbody>
</table>

∴ null hypothesis (HA0) must be accepted

### Test 2. Frequency of Logins vs Marks

<table>
<thead>
<tr>
<th>Measure</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spearman's Rank Correlation Coefficient r</td>
<td>0.372954685</td>
</tr>
<tr>
<td>Statistical Significance of Correlation Two-Tailed T Test</td>
<td>0.325</td>
</tr>
</tbody>
</table>

∴ hypothesis (HA1) can be accepted

Formula Used For Correlation Is: (Spearman's Rank Correlation Coefficient)

\[
\begin{align*}
n & = \text{number of pairs of scores} \\
\sum xy & = \text{sum of the products of paired scores} \\
\sum x & = \text{sum of x scores} \\
\sum y & = \text{sum of y scores} \\
\sum x^2 & = \text{sum of squared x scores} \\
\sum y^2 & = \text{sum of squared y scores}
\end{align*}
\]
Analysis was undertaken on the following data sets:

**Table 1. x and y scores**

<table>
<thead>
<tr>
<th>Student No.</th>
<th>eWURF Mark (x)</th>
<th>Usage (Reqs Produced) (y)</th>
<th>Usage (Num Logins) (y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>23</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>7</td>
<td>14</td>
<td>34</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>8</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>6</td>
<td>8</td>
<td>18</td>
</tr>
<tr>
<td>15</td>
<td>5</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>6</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>4</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
<td>28</td>
<td>15</td>
</tr>
<tr>
<td>21</td>
<td>7</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>22</td>
<td>6.5</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>23</td>
<td>6</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>24</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>9</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>26</td>
<td>8</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>27</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>5</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>30</td>
<td>6</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>31</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>32</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>33</td>
<td>5.5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>34</td>
<td>5</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>35</td>
<td>4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>36</td>
<td>7</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>37</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**Totals:** | 193 | 143 | 205
Table 2. sum of squared x and y scores (Test 1 and Test 2)

<table>
<thead>
<tr>
<th>(Test 1)</th>
<th>(Test 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>x*y</td>
<td>x*x</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>48</td>
<td>64</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
</tr>
<tr>
<td>12</td>
<td>36</td>
</tr>
<tr>
<td>48</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
</tr>
<tr>
<td>63</td>
<td>49</td>
</tr>
<tr>
<td>42</td>
<td>49</td>
</tr>
<tr>
<td>10</td>
<td>25</td>
</tr>
<tr>
<td>24</td>
<td>36</td>
</tr>
<tr>
<td>98</td>
<td>49</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
</tr>
<tr>
<td>48</td>
<td>36</td>
</tr>
<tr>
<td>15</td>
<td>25</td>
</tr>
<tr>
<td>18</td>
<td>36</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>112</td>
<td>16</td>
</tr>
<tr>
<td>70</td>
<td>49</td>
</tr>
<tr>
<td>52</td>
<td>42.25</td>
</tr>
<tr>
<td>24</td>
<td>36</td>
</tr>
<tr>
<td>7</td>
<td>49</td>
</tr>
<tr>
<td>45</td>
<td>81</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>36</td>
</tr>
<tr>
<td>20</td>
<td>25</td>
</tr>
<tr>
<td>6</td>
<td>36</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>49</td>
</tr>
<tr>
<td>0</td>
<td>30.25</td>
</tr>
<tr>
<td>25</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>28</td>
<td>49</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>859</td>
<td>1257.5</td>
</tr>
</tbody>
</table>
Module: WAU - Assessment Mark vs Usage Relationship Analysis

HA1: Increased usage of the CAWE tool will result in higher marks for requirements analysis.

HA0: Increased usage of the CAWE tool will not result in higher marks for requirements analysis.

<table>
<thead>
<tr>
<th>Test 1. Requirements documents produced vs Marks</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Spearman's Rank Correlation Coefficient $r =$</td>
<td>0.226230711</td>
</tr>
<tr>
<td>Statistical Significance of Correlation Two-Tailed T Test =</td>
<td>0.325</td>
</tr>
</tbody>
</table>

∴ null hypothesis (HA0) must be accepted

<table>
<thead>
<tr>
<th>Test 2. Frequency of Logins vs Marks</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Spearman's Rank Correlation Coefficient $r =$</td>
<td>0.372954685</td>
</tr>
<tr>
<td>Statistical Significance of Correlation Two-Tailed T Test =</td>
<td>0.325</td>
</tr>
</tbody>
</table>

∴ hypothesis (HA1) can be accepted

Formula Used For Correlation Is: (Spearman's Rank Correlation Coefficient)

- $n =$ number of pairs of scores
- $\sum xy =$ sum of the products of paired scores
- $\sum x =$ sum of $x$ scores
- $\sum y =$ sum of $y$ scores
- $\sum x^2 =$ sum of squared $x$ scores
- $\sum y^2 =$ sum of squared $y$ scores

431
Analysis was undertaken on the following data sets:

**Table 1. x and y scores**

<table>
<thead>
<tr>
<th>Student No.</th>
<th>eWURF Mark (x)</th>
<th>Usage (Reqs Produced) (y)</th>
<th>Usage (Num Logins) (y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>23</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>7</td>
<td>14</td>
<td>34</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>8</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>6</td>
<td>8</td>
<td>18</td>
</tr>
<tr>
<td>15</td>
<td>5</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>6</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>4</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
<td>28</td>
<td>15</td>
</tr>
<tr>
<td>21</td>
<td>7</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>22</td>
<td>6.5</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>23</td>
<td>6</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>24</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>9</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>26</td>
<td>8</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>27</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>5</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>30</td>
<td>6</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>31</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>32</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>33</td>
<td>5.5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>34</td>
<td>5</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>35</td>
<td>4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>36</td>
<td>7</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>37</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**Totals:** 193 143 205
### Table 2. Sum of squared $x$ and $y$ scores (Test 1 and Test 2)

<table>
<thead>
<tr>
<th></th>
<th>(Test 1)</th>
<th></th>
<th>(Test 2)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x \times y$</td>
<td>$y^2$</td>
<td>$x^2$</td>
<td>$y^2$</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>1</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>48</td>
<td>64</td>
<td>36</td>
<td>96</td>
<td>64</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>1</td>
<td>0</td>
<td>25</td>
</tr>
<tr>
<td>12</td>
<td>36</td>
<td>4</td>
<td>6</td>
<td>36</td>
</tr>
<tr>
<td>48</td>
<td>64</td>
<td>36</td>
<td>80</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
<td>1</td>
<td>72</td>
<td>64</td>
</tr>
<tr>
<td>63</td>
<td>49</td>
<td>81</td>
<td>49</td>
<td>49</td>
</tr>
<tr>
<td>42</td>
<td>49</td>
<td>36</td>
<td>161</td>
<td>49</td>
</tr>
<tr>
<td>10</td>
<td>25</td>
<td>4</td>
<td>5</td>
<td>25</td>
</tr>
<tr>
<td>24</td>
<td>36</td>
<td>16</td>
<td>36</td>
<td>36</td>
</tr>
<tr>
<td>98</td>
<td>49</td>
<td>196</td>
<td>238</td>
<td>49</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
<td>1</td>
<td>16</td>
<td>64</td>
</tr>
<tr>
<td>48</td>
<td>36</td>
<td>64</td>
<td>108</td>
<td>36</td>
</tr>
<tr>
<td>15</td>
<td>25</td>
<td>9</td>
<td>10</td>
<td>25</td>
</tr>
<tr>
<td>18</td>
<td>36</td>
<td>9</td>
<td>18</td>
<td>36</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>16</td>
<td>0</td>
<td>28</td>
<td>16</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>112</td>
<td>16</td>
<td>784</td>
<td>60</td>
<td>16</td>
</tr>
<tr>
<td>70</td>
<td>49</td>
<td>100</td>
<td>70</td>
<td>49</td>
</tr>
<tr>
<td>52</td>
<td>42.25</td>
<td>64</td>
<td>19.5</td>
<td>42.25</td>
</tr>
<tr>
<td>24</td>
<td>36</td>
<td>16</td>
<td>42</td>
<td>36</td>
</tr>
<tr>
<td>7</td>
<td>49</td>
<td>1</td>
<td>0</td>
<td>49</td>
</tr>
<tr>
<td>45</td>
<td>81</td>
<td>25</td>
<td>81</td>
<td>81</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
<td>1</td>
<td>8</td>
<td>64</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>36</td>
<td>0</td>
<td>0</td>
<td>36</td>
</tr>
<tr>
<td>20</td>
<td>25</td>
<td>16</td>
<td>20</td>
<td>25</td>
</tr>
<tr>
<td>6</td>
<td>36</td>
<td>1</td>
<td>6</td>
<td>36</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>49</td>
<td>1</td>
<td>0</td>
<td>49</td>
</tr>
<tr>
<td>0</td>
<td>30.25</td>
<td>0</td>
<td>27.5</td>
<td>30.25</td>
</tr>
<tr>
<td>25</td>
<td>25</td>
<td>25</td>
<td>30</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>1</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>28</td>
<td>49</td>
<td>16</td>
<td>42</td>
<td>49</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>859</td>
<td>1257.5</td>
<td>1549</td>
<td>1329</td>
<td>1257.5</td>
</tr>
</tbody>
</table>
Appendix C5 – Student Generated SRS Document

Online Portfolio

Document Version: 6

Date Of Issue: Tuesday 26th April 2010
Document Author: xxxxx xxxxxx  |  xxxxxx@tees.ac.uk

Project Description: An online system that will allow users to register and login to the website. They can then upload their cv and images and view all uploaded content. There will also be an administration section.

Statement Of Purpose: I will develop a website using a mixture of HTML, CSS, PHP and mySQL. The user interface of the website will be designed in HTML. Server side scripts will be developed using PHP to produce dynamic objects. mySQL will be used to create database tables, and CSS will be used to define the layout and appearance of the website. SQL queries will also be implemented into the PHP code. The purpose is to develop functionality that will allow a user to register and then log in to the website. The member can then upload images of themselves along with information from their curriculum vitae. The website will include a section to update any information stored about the member and an option to edit or delete their curriculum vitae and image. The website will also include an Administrators section that will allow an administrator to log in to the system and view and edit all information from all user accounts.

Objective1: To help job seekers promote themselves to potential employers

Objective2: Improve my PHP skills

Objective3: Improve Photoshop skills

Objective4: Understand SQL queries in further detail
Objective 5: To allow general browsers to view the website but not secure content

Objective 6: To allow people to register and upload information from their CV and include images

Tasks And Associated Actors

Task Name: Create a registration form that will require email activation  Actor: Developer,
Task Name: Create a log in section  Actor: Developer,
Task Name: Create a log out section  Actor: Developer,
Task Name: Create a section that will allow a user to upload an image of themselves  Actor: Developer,
Task Name: Create a section that will allow a user to update any details stored about themselves  Actor: Developer,
Task Name: Create a section that will allow a user to upload their CV  Actor: Developer,
Task Name: Create an administration section  Actor: Developer,
Task Name: Create a section that allows a user to edit and delete all information and images for their own user account  Actor: Developer,

Functional Req’s And Associated Actors

Functional Req's: Shall allow people to register. Shall not allow people to access secure area without logging in. Must allow people access with the correct credentials.  Actors: General browsers of the website, Person seeking employment (Upload their CV),
Functional Req's: Must allow users to upload information, cv and images. Must not allow unauthorised users access to information. Must not allow unauthorised users to edit information.  Actors: General browsers of the website, Person seeking employment (Upload their CV), Administrator,
Functional Req's: Must allow registered users to log in to the system. Must not allow unregistered users to log in to the system. Must only accept the correct username and password combination.  Actors: General browsers of the website, Administrator, Person seeking employment (Upload their CV),
## Web Requirements

**Key:** FR-x = Functional Requirement || NFR-x = Non-Functional Requirement

<table>
<thead>
<tr>
<th>FR-1</th>
<th>Functional Requirement 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>functional requirement</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description:</strong> Shall allow people to register. Shall not allow people to access secure area without logging in. Must allow people access with the correct credentials.</td>
<td></td>
</tr>
<tr>
<td><strong>Associated Actors:</strong> General browsers of the website, Person seeking employment (Upload their CV), , ,</td>
<td></td>
</tr>
<tr>
<td>These Actors will need to be taken into consideration when designing/developing this requirement!</td>
<td></td>
</tr>
<tr>
<td><strong>Comments:</strong> Registration</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FR-2</th>
<th>Functional Requirement 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>functional requirement</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description:</strong> Must allow users to upload information, cv and images. Must not allow unauthorised users access to information. Must not allow unauthorised users to edit information.</td>
<td></td>
</tr>
<tr>
<td><strong>Associated Actors:</strong> General browsers of the website, Person seeking employment (Upload their CV), Administrator, ,</td>
<td></td>
</tr>
<tr>
<td>These Actors will need to be taken into consideration when designing/developing this requirement!</td>
<td></td>
</tr>
<tr>
<td><strong>Comments:</strong> Portfolio</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FR-3</th>
<th>Functional Requirement 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>functional requirement</strong></td>
<td></td>
</tr>
</tbody>
</table>
Description: Must allow registered users to log in to the system. Must not allow unregistered users to log in to the system. Must only accept the correct username and password combination.

Associated Actors: General browsers of the website, Administrator, Person seeking employment (Upload their CV), .
These Actors will need to be taken into consideration when designing/developing this requirement!

Comments: Log In

NFR-1
User Interface
Non-Functional Requirement
Description: The User Interface will be designed at a screen resolution of 800x600. The layout will be Fixed and the User Interface will scroll if there is any excess content. The main text size will be Scaleable. The main navigation will be of Top type and will use Bread Crumb navigational aids to help the user see where they are within the application.

Associated Actors: Not Applicable To Non-functional Requirements

Additional Req's: null

NFR-2
Marketing
Non-Functional Requirement
Description: The website will use Domain Name for Search Engine Optimisation. In order to track the metrics of any marking campaigns, the site will use in order to evaluate it effectiveness in terms of traffic. In terms of website visibility and rankings on search engines, we will use Manual Submission in order to achieve this. In order to evaluate goal conversion we will use the following; Review in house.

Associated Actors: Not Applicable To Non-functional Requirements

Additional Req's: Website hosted on Universities internal web server so search engine optimisation is not applicable.
NFR-3

Accessibility

Non-Functional Requirement

Description: The Accessibility of the site will be enhanced by meeting Priority 3 WAI Guidelines. The following assistive technology will be used: ALT Tags Applied To Images.

Associated Actors: Not Applicable To Non-functional Requirements

Additional Req's: null

NFR-4

Usability

Non-Functional Requirement

Description: The type of website that is being designed requires the Efficiency of tasks to be: Very Important, the Learnability to be: Medium Importance, and Memorability of tasks to be: Very Important. The site will use Basic searching to help users find information on the site. It is required that that loads in less than 10 seconds.

Associated Actors: Not Applicable To Non-functional Requirements

Additional Req's:

NFR-5

Technical

Non-Functional Requirement

Description: The development team must use PHP and MySQL as the serverside technology. The server will therefore be: Linux.

Associated Actors: Not Applicable To Non-functional Requirements

Additional Req's: null