VuVote
‘Vic Uni’s Online Voting System’
Design Project Report
(DPR)

28th October 2002

Developer: Gary Howells

Client: Cameron Phillips

Supervisor: Dan Nelson
III. TABLE OF CONTENTS

TABLE OF FIGURES ........................................................................................................................................... 3

1. INTRODUCTION ........................................................................................................................................ 7

2. DESIGN PROCESSES ....................................................................................................................... 7

3. PROJECT MANAGEMENT .............................................................................................................. 9

4. PROJECT EVALUATION .................................................................................................................... 10

5. CONCLUSION ........................................................................................................................................ 12

6. BIBLIOGRAPHY ..................................................................................................................................... 13

7. DESIGN APPENDICES ........................................................................................................................... 14

  7.1 Design Appendix A – Project Management .................................................................................. Error! Bookmark not defined.

  7.2 Design Appendix B – Problem Statement (PS) ........................................................................... Error! Bookmark not defined.

  7.3 Design Appendix C – Software Requirements Specification (SRS) ........................................ Error! Bookmark not defined.

1. Introduction............................................................................................................................................. Error! Bookmark not defined.

2. Existing System ..................................................................................................................................... Error! Bookmark not defined.

3. Proposed System ................................................................................................................................... Error! Bookmark not defined.

4. Functional Requirements .................................................................................................................. Error! Bookmark not defined.

5. Non-Functional Requirements ........................................................................................................ Error! Bookmark not defined.

5.1 User Interface and Usability Requirements .................................................................................... Error! Bookmark not defined.

5.2 Performance Requirements ........................................................................................................... Error! Bookmark not defined.

5.3 Operational Requirements ............................................................................................................. Error! Bookmark not defined.

5.4 Maintainability and Portability Requirements .............................................................................. Error! Bookmark not defined.

5.5 Security Requirements .................................................................................................................... Error! Bookmark not defined.

6. Acceptance Criteria ............................................................................................................................. Error! Bookmark not defined.

7. Glossary ................................................................................................................................................ Error! Bookmark not defined.

Bibliography ............................................................................................................................................. Error! Bookmark not defined.

References ................................................................................................................................................ Error! Bookmark not defined.

Client Sign-off ........................................................................................................................................ Error! Bookmark not defined.

7.4 Design Appendix D – Software Design Document (SDD) ........................................................ Error! Bookmark not defined.

1. Introduction............................................................................................................................................. Error! Bookmark not defined.

2. Design Development ........................................................................................................................ Error! Bookmark not defined.

3. Architectural Design ........................................................................................................................ Error! Bookmark not defined.

4. Detailed Design..................................................................................................................................... Error! Bookmark not defined.

4.1 Database Design .............................................................................................................................. Error! Bookmark not defined.

4.2 Administrator Web-Page Design .................................................................................................... Error! Bookmark not defined.

4.3.1 Registered User: Eligible Screen .............................................................................................. Error! Bookmark not defined.

4.2.0 Login .............................................................................................................................................. Error! Bookmark not defined.

4.2.1 Administrator: Admin Main ....................................................................................................... Error! Bookmark not defined.

4.2.2 Administrator: Eligible Election .............................................................................................. Error! Bookmark not defined.

4.2.3 Administrator: Create Election ............................................................................................... Error! Bookmark not defined.

4.2.4 Administrator: Update Election ............................................................................................. Error! Bookmark not defined.

4.2.5 Administrator: Delete Election ............................................................................................... Error! Bookmark not defined.

4.2.6 Administrator: Confirm Election Details ................................................................................ Error! Bookmark not defined.

4.3 User Web-Page Design .................................................................................................................. Error! Bookmark not defined.

4.3.2 Registered User: Election Outline ............................................................................................ Error! Bookmark not defined.

4.3.3 Registered User: Candidate Information ................................................................................ Error! Bookmark not defined.

4.3.4 Registered User: Ballot Form .................................................................................................... Error! Bookmark not defined.

4.3.5 Registered User: Issue Receipt .................................................................................................. Error! Bookmark not defined.

4.4 Static Web-Page Design ................................................................................................................ Error! Bookmark not defined.

4.4.1 Static Information: How to Vote .............................................................................................. Error! Bookmark not defined.

4.4.2 Static Information: About VuVote .......................................................................................... Error! Bookmark not defined.

4.4.3 Static Information: Not able to Vote? ..................................................................................... Error! Bookmark not defined.

4.4.4 Static Information: Contact Information ............................................................................... Error! Bookmark not defined.

4.4.5 Static Information: Privacy Issues .......................................................................................... Error! Bookmark not defined.

4.4.6 Static Information: Boards / Committees .............................................................................. Error! Bookmark not defined.

4.5 Algorithmic Architecture Design ............................................................................................... Error! Bookmark not defined.
TABLE OF FIGURES

Figure 1: - Timeline Breakdown ................................................................. 9
Figure 2: - Time Spent Graph (hrs) ............................................................. 10
Figure 3: Use Case Diagram: Administrator ............................................. Error! Bookmark not defined.
Figure 4: Use Case Diagram: Unregistered User ....................................... Error! Bookmark not defined.
Figure 5: Use Case Diagram - Registered User ......................................... Error! Bookmark not defined.
Figure 11: Table of Sizes ........................................................................ Error! Bookmark not defined.
Figure 12: - Architectural Design ............................................................. Error! Bookmark not defined.
I. ACKNOWLEDGEMENTS

I would like to thank the following people for their support in the design and development:

Thank you to:

- Terry-Ann Joy for the understanding of the school record structure, and supplying various IT contacts.
- Russel Woodley for time taken describing the previous attempts of electronic voting at Victoria University.
- Bob Ritchens for discussions to the process of how an election in finalised.

With Special Thanks to:

- Peter and Edna Howells for their emotional support over the duration of this design project.
- Alisha Wilkins for her emotional support over the duration of this design project.
- Cameron Phillips for his support of the AusVote system, I look forward to additional dealings with Cameron in the New Year.
- Daniel Nelson for his help as a supervisor and mentor throughout the Design Project period, and going beyond his duties as a supervisor to ensure that this design was worthwhile.
- Elizabeth Haywood for her on going help, and support throughout the year in the development of the AusVote system.
- Jane Madden for the consistent excellence in report correction and suggestions for improvement.
- Steven de Bruyn for his various joint efforts such as mail, and LDAP connectivity.
- Cameron Giles for modification to the Melba server, and various technical support throughout the year.
- Marjanco Todoroski, Selcuk Samsa, and Christopher Perry for their support with key features of design.
- Stewart James for his advice and technical support with the VU LDAP server.
II. SYNOPSIS

Victoria University (VU) runs elections throughout the year to fill vacancies on their various boards and committees. These are done manually taking the time of both the staff and voting officials. As VU hosts twelve campuses, large elections take weeks to conduct. Incorporated into the running of an election is the manual counting of votes, and the mountains of paper waste.

By introducing the new Internet based system, allowing both staff and students to be able to vote at their own leisure, time and paper waste can be avoided. The new system named ‘AusVote’ will allow a user to logon to the system, using their VU mail username and password, read relevant candidate policy statements, and vote for their preferred candidate in a manner outlined by Statute 8.1.1 – Election Procedures.

This system will be available to 100,000 students at 5,000 staff, and although this is the current number of staff and students at Victoria University it does not have a bound on this number.

Since the database used is MySQL, large numbers of ‘voters’ can use the system at the same time, this is only bound by the number of simultaneous connections outlined by the nominated VU server, which has not been determined at this time (see Software Specifications Document).

‘AusVote’ will aim to service the 105,000 plus ‘voters’ allowing storage for votes, calculation of successful candidates, statistics of voters primary campus, and allow for a knowledgeable vote by providing online candidate policies. At the same time, ‘AusVote’ will ensure that the candidate may vote only once per election, that a ‘voters’ preference is confidential not being linked back to them in any way, and provide navigational tools and help for novice users.

This product was completed using Software Engineering principles and guided by the Software Development Life Cycle (SDLC) to produce several supporting documents. The first of these documents was a Problem Statement (see appendix B-I) that outlined the current system implemented at Victoria University, and a proposal a new computer based system named AusVote. After identifying the problem to be computerized, a Software Requirement Specification (see appendix B-II) document was produced to ensure that the product adhered to the specifications outlined by the client Cameron Phillips. Next a Software Design Document (see appendix B-III) was produced to outline how the product would be produced and show the functionality between each webpage, and the database. A Test Plan (see appendix B-IV) was also created to outline the test cases for each webpage to ensure the correct working of each page. Finally a Design Project Report (this document) was created to ensure that the system adheres to the initial specifications, and outline the project as a whole.

This product complies with the bound specifications outlined in the Software Specifications Document (see appendix B-II) including the desired optional functionality of counting the number of voters per campus, and the ability to total the votes per election to determine the winner.

In total about 450 hours were spent developing the project the from initial problem statement through the software development life cycle to the finalisation of the project. Like many software projects, after the testing phase is complete, many different cosmetic improvements can be made, it is at this point that the project must be deemed complete otherwise it could be never ending.

The ‘AusVote’ system is to be trialled at the beginning of 2003, to demonstrate the correct functionality of the system, and will be accompanied by the traditional paper-based system for a term nominated by Cameron Phillips until confidence of the product is established.

After successful test, additional documents will need to be produced to ensure the correct maintenance of the AusVote system.
1. INTRODUCTION

Victoria University (VU) has a manual system of conducting and tallying votes for in-house elections. Elections are run throughout the year. Several elections can be run at the same time, determining successful candidates for particular Board positions. As Victoria University hosts fourteen campuses large elections take weeks to conduct wasting the time of the several officials that could be better spent on other duties.

The proposed global system named ‘VuVote’ would allow staff and students (‘voters’) to vote online. This system will provide information on candidates, the VU voting system and other information through an easy to use, graphical website, which will not only decrease voter frustration, but increase the efficiency and participation in the voting process.

The current VU voting system asks each member of staff to give nominations for a particular election. Nominations are then placed on the Ballot in random order, a Ballot Paper is sent to each eligible staff member. After filling in their preferences a Ballot Paper is returned to the Election Official to be counted.

This method requires not only the time of the voter to read through and make a selection of their preferred candidate, but it also takes the time of the various officials who must count each of the Ballot Papers in accordance with University regulations (Statute 8.1.1 – Election Procedures).

As there are currently 80,000 enrolments and 5,000 staff members, the possible number of candidates could be up to 100,000. This causes a major problem when conducting large elections, as not all candidates will be known University wide. The VuVote system would allow a voter to gather information on candidates, boards/committees, and election information to make a knowledgeable vote levelling the playing field for all candidates.

The system must conform to a list of requirements (both functional and non-functional) that are outlines in accompanying documents.

The purpose of this report is to present an overview of this Design Project for the VuVote system. This will include the process in which the system design was undertaken, project management (including time allocation) and an evaluation of the project as a whole. This will demonstrate how the solution was reached, and the method that was undertaken to solve this software engineering project.

2. DESIGN PROCESSES

The VuVote system was created through several meetings with the project supervisor, Dan Nelson, to discuss the possibilities of making the system more viable to Victoria University. A client was found within the University, Cameron Phillips (), who was interested in making the system online to increase the number of voters for each election. Cameron felt that if the system were created online, then staff and students would be able to vote at their own leisure, and if additional information were given about candidates then the chance of electing a worthy candidate would be more likely.

Before meeting with Cameron background information was sought into the voting process from both the standpoint of the voter, in this case Dan Nelson, who described through the process that must be
undergone to vote in particular elections (As described in appendix B – Problem Statement). The current system was also reviewed from the viewpoint of another voting official, Bob Ritchens, who described the process of how each election was run (From start to finish).

Now having preliminary information on the VU voting process, a meeting with Cameron took place to discuss the method undertaken to find a successful candidate. At this time discussions took place about stakeholders of the system, these being the voters, Administrators, and general public, and their various roles, included in this discussion was Cameron’s list of wants and needs (functional and non-functional requirements) of the system.

To commence a project of this magnitude several specification documents were written to analyse the problem at hand. The first of these documents was the Problem Statement; this document outlined the various stakeholders in the system and the problems related to each, it outlines the roles that each stakeholder interacts with the system and how the introduction of an automated system would be beneficial.

Cameron’s list of functional requirements was as follows:

- A voter must only be able to vote once.
- The vote must be confidential and must not be linked to the voter in any way.
- The system should include security to minimise the risk of outside influence.
- A voter must be able to vote from anywhere with the necessary connection to the Internet.
- The system must be able to tally votes at the end of the election (desirable).
- If a voter feels they should be eligible for an election but cannot vote, they must be able to contact Cameron to rectify the problem (desirable).
- Statistics on the number of voters per campus (desirable).
- The voter can use any type of computer to access the system (desirable).

Through listing the above requirements Cameron agreed to be the client for the VuVote system.

After finding the basic requirements for the system large amounts of research were needed, this included research into the best language, database package, and format for the system design.

A similar system has already been trailed by with mixed results from student union, after discussing the issue of Online Voting with the union it came to light that an external company created their system but it was only a simple database program that required that computers were brought in specially for each election, hence their voting numbers were still low.

The system therefore must be available on the Internet, and include various security features. The database must not allow the Voter’s unique identification to be matched up with their vote in any way, and also have security features to ensure external tampering cannot take place.

From this research came a complete Software Requirements Specification (Appendix C - SRS) that outlines in detail the product scope (user scenarios), an elaboration of the Functional and Non-Functional requirements, and an estimate of the size of the final product including a data structure map. It was at this stage that test code took place to understand the chosen language, PHP.

PHP was used because unlike many other languages it required no installation on the host computer (i.e. Voter), and due to this it was the ideal candidate to use. PHP also had another advantage, it came with built in security as no code can be seen when trying to view the source from a browser. In addition to PHP, a MySQL database was used to hold required data (i.e. Election Dates, Candidate information etc.) and HTML was used to display basic features, such as menu items.

The specifications outlined in the SRS allowed additional in-depth research into authentication protocols, and programming techniques.

After additional research was undertaken additional documentation, in the form of a Software Design Document (SDD), was required to outline the interactions between the database (MySQL) and the Web Pages (HTML). Also in this document is the relational schema that describes the interaction between
each of the database tables, and also shows the transition between each of the web pages including what a user must do to navigate between them (refer to appendix D).

At this time pseudo-code was produced to describe the interaction, using PHP, between the web pages and the database. Although only a basic outline of the code at this stage it shows the large extent of the code to come. At this stage screenshots were produced for each page to show what the final outcome should look like, these were just a guideline, and, as a running document will change over time.

After completing the SDD the coding took place, this meant coding both the database and the web pages to match the specifications. First the database was implemented and links between the tables were made. When completed the basic outline of the HTML pages formed, creating the basis of the assignment to come. At this stage 140 hours was spent on the project, which included meeting times, and documentation writing. Coding took place for the following five weeks, and although many hours were put into the development and design of the system over this period, it is not yet complete.

At this stage a Test Plan was created to ensure that the information past from each page and the data entered will be correct on completion, and after following the test procedures for each of the modules the information will be redeemed correct.

The Test Plan (TP) has a test case for each of the pages outlining what it should and should not do under certain conditions (refer to Appendix E).

3. PROJECT MANAGEMENT

The completion of ‘VuVote’ (the Online Voting System) was expected to take around 300 hours, this would allow a software package that was well planned, developed, and bug free. However, this number quickly grew and now as the project comes closer to completion the estimated time of finish will be approximately 450 hours. Thus, the approximate finish time will be 50% more than that of the estimate.

Although this difference is quite large (approx 150hrs) this allowed greater attention to detail, and more client control when producing the final product. Due to a large amount of time dedicated to the project (17.8hrs per week) that will allow the completion on time including all initial specifications, requirements, and suggestions for the next version of the software.

The recorded time for this project can be split up into six different sub-groups and showing the times (in hours) respectively:

<table>
<thead>
<tr>
<th>Topic Undertaken</th>
<th>Time Spent (hrs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Research</td>
<td>22</td>
</tr>
<tr>
<td>Code</td>
<td>296</td>
</tr>
<tr>
<td>Documentation</td>
<td>120</td>
</tr>
<tr>
<td>Meetings</td>
<td>34.5</td>
</tr>
<tr>
<td>Presentation and Preparation</td>
<td>20.5</td>
</tr>
<tr>
<td>Algorithms</td>
<td>56</td>
</tr>
</tbody>
</table>

Figure 1: - Timeline Breakdown
Research includes all aspects of the project relating to in depth study of LDAP, MySQL, PHP, etc, but does not include items such as meetings with IT and Technical Support from various sources. Additional technical research was also undertaken whilst coding the product.

The Coding of the project was the most difficult as learning a software language is difficult, but learning a language whilst coding the project simultaneously was extremely difficult, and took the most time. This project was written in raw HTML and PHP in a text editor, and did not have aid of any precompiled drag-and-drop website development tools. Although this is more challenging than using a precompiled editor it allows greater control of the final ‘look and feel’ of the product.

Documentation took a large amount (almost 31%) of the undertaken project time, and although this time was significant, it allowed for strong foundations in the design process. Having these foundations in place reasonable functional and non-functional specifications were produced and agreed upon, allowing for smooth communication between client, supervisor and developer.

Meetings played an important role in the development of the VuVote software. This allowed feedback from client into the functionality of the software, correct terminology, general information and layout of the design. The supervisor gave feedback on the quality of the documentation, project and time management. Meetings took place with the supervisor weekly for approximately one hour per week (not all recorded in log book) and with the client once per month. Although the client was not consulted as regularly as the supervisor communication via email often took place.

Presentations and Preparation took place throughout the year to demonstrate the working ability of the software in front of various sized audiences. Preparation for these demonstrations took place by creating talks, and slides to support the presentation. Throughout the year to date there have been three talks, with a final demonstration due at the end of the year in front of a large audience.

The other largely time consuming task was to develop the various Algorithms needed for the Online Voting Software to function correctly. Initially guidelines to the process undertaken to produce an overall successful candidate for a current election was sought from the client, Cameron Phillips, in an effort to follow the current procedures of Victoria University (As outlined in University Statute 8.1.1 – Election Procedures and discussed in the included Software Design Document). The Algorithms included the method that the database was updated, and the fashion that the individual ballot was conducted and stored for later recollection.

To view a breakdown of the number of hours taken for each Topic see Appendix A1.

4. PROJECT EVALUATION

Although this software package started out as a demonstration of what was possible to create for Victoria University, it has since been learnt that due to the time spent developing the Online Voting
Software and the quality of the final product, that the software will be tested, and put to use at Victoria University under trial conditions.

The software has met the requirements outlined by the Client, Cameron Phillips, in the Software Specifications Document (SRS) in full, including optional components deemed as ‘desires’ rather than requirements.

The solution has not yet been fully tested to ensure the functional correctness of the product, but will be tested against the list of test cases outlined in the Test Plan (TP), and after which will be deemed functionally correct and ready for trial use, only after passing the trial stage by Cameron Phillips shall the product ‘VuVote’ be deemed a successful solution.

Various problems arose during the creation of the software solution, which included security issues, time restraints, and communications issues. These issues were solved using several methods, and through discussions with various people.

Security Issues included passing variable between pages for reference with the MySQL database, various PHP functions and the LDAP information server. This was overcome using sessions, which allows information to be passed through each page encrypted, the information is stored in a session key that resides in the MySQL database.

The database for secure login was first used as a MySQL table, but this only allowed the small number of people residing in the database to sign into the Web page, after much consideration it was found that VU hosted an LDAP server that held the record of each student and staff member including relative records. LDAP was therefore used to confirm the username and password for each voter and return the campus, their email address, and unique username for later use.

Due to finding these issues at the beginning of the project, they were easily overcome. It built on the strength of the product, and allowed security flaws to be fixed.

The VuVote system uses advanced security using both sessions, to ensure that users cannot transverse between pages without logging in, and the VU LDAP server, that allows only current staff and students to access the VuVote system that is maintained by student services.

However, VuVote does have some limitations, when conducting a ballot the number of recorded votes for that ballot is restricted to 20 votes, which in the unlikely case of having more than 20 candidates running in a particular election, and a voter number the votes 1 to 21+ votes; any votes after 20 are disregarded to maintain database integrity.

VuVote does have other small limitations that will be overcome in later versions of the software. These include not automatically adding the successful candidate to the board they are successful to join. However, the administrator, Cameron Phillips, does have the option, included in the current release to enable him to add the successful candidate using a predefined form for eligible voter modification.

The second version of the software will include a running output of the votes to ensure vote integrity by copying a running total of the votes to read-only media (such as CD) to combat administrator scrutiny from disgruntle candidates, and modification at database level from the administrator. This was not a functional requirement, but was discussed as an essential feature of the next version of the software.

In the next version the colour scheme of the website will be changed to match that of the current VU websites, this is to allow the voting set to recognise the Website as part of the VU set, and hence comfort the voter when using the VuVote product.

Another means of entering the candidate’s information will be sought, this maybe possible in a number of manners, the first would be to allow candidate to administrate their own policy, however, this would not be a best solution in the case of derogatory or criticising view points, which may offend other voters. Currently in the manual system each candidate is to provide their policy in a word document to Cameron, who then formats it and puts it with the ballot paper in random order, this could be modified to allow the policy to be retrieved from the word document that would have the election number
followed by the username of the candidate for the election. In the current specification of the VuVote system, the administrator must manually set each candidate, copying his or her policy into the database.

Nominations will still be conducted by hand due to the need of dual signatures (2) from eligible voters, and a confirmation from the alleged candidate, this system would then be replicated online and given a two-week grace period to call for candidates.

Several lessons have been learnt over this six-month period, including the value of communication, meeting structure, project and time management. These will be extremely useful in future assignments and aspirations. This project has also shown the importance of good software engineering design principles, which have allowed the project to develop on time, and to develop smoothly.

5. CONCLUSION

This design has been created from requirements discussed with the client Cameron Phillips. It has since been developed through each phase of the software lifecycle to produce a working software package to be trailed at Victoria University (VU) for their online voting; taking the system from a paper-based manual system to an online computer system that stores each vote for a particular election, and tallies the vote when an election has ended.

For this design to mature and develop, it needed to keep a focus on the functional and non-function requirements of the system and abide by the specifications initially laid out my Mr. Cameron Phillips that outline the ‘working’ of the VuVote system.

To ensure that this design is complete in all aspects and is ready for implementation on one of the many VU servers, as defined in the Software Requirements Specification, many hours were poured into the functionality of the system, and how it would preform under various circumstances and tested using test-cases outline in the test plan document (accompanying this one).

After successful test, several other documents will need to be produced to ensure the correct maintenance of the VuVote system.

I would personally like to thank Cameron Phillips for the opportunity to create this software on his behalf, and show my great delight that this system will be trialed over the next few months to show its correct functionality. I look forward to dealings with Cameron in the future.
6. BIBLIOGRAPHY


7. DESIGN APPENDICES