ReMoTe: A complete tool to support software process management

Darrion Todd DeMelo

Follow this and additional works at: https://scholarworks.lib.csusb.edu/etd-project

Part of the Software Engineering Commons

Recommended Citation
DeMelo, Darrion Todd, "ReMoTe: A complete tool to support software process management" (2006). Theses Digitization Project. 3104.
https://scholarworks.lib.csusb.edu/etd-project/3104

This Project is brought to you for free and open access by the John M. Pfau Library at CSUSB ScholarWorks. It has been accepted for inclusion in Theses Digitization Project by an authorized administrator of CSUSB ScholarWorks. For more information, please contact scholarworks@csusb.edu.
ReMoTe: A COMPLETE TOOL TO SUPPORT SOFTWARE PROCESS MANAGEMENT

A Project
Presented to the
Faculty of
California State University,
San Bernardino

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Computer Science

by
Darrion Todd DeMelo
March 2006
ReMoTe: A COMPLETE TOOL TO SUPPORT SOFTWARE PROCESS MANAGEMENT

A Project
Presented to the
Faculty of
California State University,
San Bernardino

by
Darrion Todd DeMelo
March 2006

Approved by:

Arturo I. Concepcion, Chair, Computer Science

David Turner

George Georgiou
ABSTRACT

When dealing with software development, every company must organize and display their software development progress in order to get the project on time. The Recursively Estimating Multi-Threaded Observation Technology Enterprise (ReMoTe) is a tool that addresses this issue and helps companies sort their artifacts in any field (hardware or software) in a manner that is structured and will give a progress report of the development of the project over the Web.

Originally an object oriented life-cycle philosophy, the Recursive Multi-Thread (RMT) was developed by Scott Simon and Sunny Lin. It also included multiple database support by Yi-Chen Kuo and later in the ReMoTe by Summer Xia. ReMoTe is an advanced tool of RMT that allows managers at every level of the project organization to effectively manage the productivity of their teams. It allows people to manage software artifacts on the ReMoTe using database systems: mySQL, Microsoft Access, or Oracle. The ReMoTe can also support the following functionalities:

1. Tracking the progress of software development based on any software life-cycle model.
2. Storage and retrieval of all software artifacts from software requirements specification to source codes.

3. Assignment and management of tasks of all personnel involved in the software project.

4. Traceability of software requirements specs from design to coding.

5. Link to common tools that the company uses, such as debugging, messaging, and chat rooms.

6. Support of multiple component software projects being developed in geographically different locations.
ACKNOWLEDGMENTS

I would like to thank Dr. Arturo I. Concepcion for giving me this opportunity to work on this tool. I would also I would like to thank him for providing me support and suggestions to evolve this project to new heights.

I would also like to thank Dr. David Turner for giving me a better perspective of Web services and how Web software is developed.

Furthermore, I would like to thank Dr. George Georgiou for giving an outside perspective of the tool and advice for the development of this project.

In addition, I would like to thank my teammates Joriz DeGuzman, Summer Xia, David Hollingsworth, Charles “Chaz” Lee, and Norman Loenandi for their help in the development, debugging and testing of this project.

Finally, I would like to express thanks for the support of the National Science Foundation under award #9810708.
# TABLE OF CONTENTS

ABSTRACT .................................................................................. iii

ACKNOWLEDGMENTS ................................................................. v

TABLE OF CONTENTS ............................................................... vi

LIST OF FIGURES ................................................................... xv

CHAPTER ONE: INTRODUCTION

1.1 History ................................................................................. 1

1.2 Purpose of the Project ...................................................... 2

1.3 Improvements to Recursive Multi-Thread ....................... 3

CHAPTER TWO: SOFTWARE REQUIREMENTS SPECIFICATION

2.1 Scope ................................................................................... 5

2.1.2 Definitions, Acronyms, and Abbreviations ................. 5

2.1.3 Overview ........................................................................... 7

2.2. Overall Description ............................................................ 7

2.2.1 Product Perspective .......................................................... 7

2.2.2 System Interfaces .............................................................. 7

2.2.3 User Interfaces ................................................................. 9

2.2.4 Hardware Interfaces ......................................................... 10

2.2.5 Software Interfaces .......................................................... 10

2.2.5.1 Communication Interface .......................................... 11

2.2.5.2 Memory Constraints .................................................. 11

2.2.5.3 Operations ................................................................. 11

2.2.5.4 Site Adaptation Requirements ................................... 12

2.2.6 Product Functions ............................................................ 12
2.3.2.5 Manage Thread Interface...... 32
  2.3.2.5.1 Approve/Deny Thread... 33
2.3.2.6 User Communication
    Interface....................... 34
  2.3.2.6.1 Message Board....... 34
  2.3.2.6.2 Chat Room.......... 36
  2.3.2.6.3 Set Task/Note........ 37
2.3.2.7 Viewing Project Interface.... 39
  2.3.2.7.1 Gantt Chart.......... 41
2.3.3 Functions ....................... 43
2.3.4 Performance Requirements ......... 43
2.3.5 Logical Database Requirements .... 43
2.3.6 Design Constraints ............... 43
  2.3.6.1 Standards Compliance .... 43
2.3.7 Software System Attributes ........ 43
  2.3.7.1 Reliability .............. 43
  2.3.7.2 Availability .......... 43
  2.3.7.3 Security ............... 43
  2.3.7.4 Maintainability ......... 43
  2.3.7.5 Portability .......... 44
2.3.8 Organizing the Specific
    Requirements .................... 44
  2.3.8.1 System Mode ............ 44
  2.3.8.2 User Class ............ 44
  2.3.8.3 Objects ............... 44
  2.3.8.4 Feature ............... 44
CHAPTER THREE: SYSTEM AND SOFTWARE DESIGN

3.1 MVC Architecture Design ......................... 45

3.1.1 Controller ................................... 47

3.1.2 Administrator Action Files ................. 48

3.1.2.1 AcceptUserAction ....................... 48

3.1.2.2 CreateProjectAction ...................... 48

3.1.2.3 DeleteMessageAction ....................... 48

3.1.2.4 DeleteProjectAction ....................... 48

3.1.2.5 DeleteUserAction ......................... 48

3.1.2.6 EditMessageAction ......................... 48

3.1.2.7 EditUserAction ........................... 48

3.1.2.8 EditUserPageAction ....................... 49

3.1.2.9 LoginAction ................................ 49

3.1.2.10 RegistrationAction ....................... 49

3.1.2.11 SetProjectManagerAction ............... 49

3.1.2.12 SetViewOnlyAction ....................... 49

3.1.3 User Action Files ............................ 50

3.1.3.1 AddThreadAction .......................... 50

3.1.3.2 ApproveThreadAction ...................... 51

3.1.3.3 CostAction ............................... 51

3.1.3.4 CostAnalysisAction ....................... 51
3.1.3.5 CreateThreadAction ............ 51
3.1.3.6 DefineThreadAction ............ 51
3.1.3.7 DeleteFileAction ............ 51
3.1.3.8 DeleteMethodAction ............ 51
3.1.3.9 DeleteNoteAction ............ 51
3.1.3.10 EditMethodAction ........... 51
3.1.3.11 EditNoteAction ............. 51
3.1.3.12 EditUserAction ............. 52
3.1.3.13 ForgotPasswordAction ........ 52
3.1.3.14 GanttChartAction ........... 52
3.1.3.15 HoursAction ................. 52
3.1.3.16 LoginAction ................. 52
3.1.3.17 MessageBoardAction ........... 52
3.1.3.18 MessageCheckAction ........... 52
3.1.3.19 NoteAction .................. 52
3.1.3.20 NoteListAction ............. 52
3.1.3.21 OverridePhaseNamesAction ...... 53
3.1.3.22 RegistrationAction ............ 53
3.1.3.23 SelectTeamAction ............ 53
3.1.3.24 SetDatesAction ............... 53
3.1.3.25 SetMethodAction ............. 53
3.1.3.26 SetNumberPhasesAction ....... 53
3.1.3.27 SetPictureAction ............ 53
3.1.3.28 SetProjectWeights ............ 53
3.1.3.29 SettingsAction ............... 53
3.1.3.30 ViewMessageAction .......... 54
3.1.3.31 ViewMessagesAction .......... 54
3.1.3.32 ViewNotesAction .......... 54
3.1.3.33 ShowFileAction .......... 54
3.1.3.34 ShowItem ................. 54
3.1.3.35 ShowTeamListAction .......... 54
3.1.3.36 StatisticsAction .......... 54

3.1.4 Interface Files ..................... 55

3.2 Database Design ..................... 58
3.2.1 User Table ................ 63
3.2.2 Uniform Resource Locator Table .... 65
3.2.3 Threads and ThreadBlob Table .... 66
3.2.4 Settings Table .............. 68
3.2.6 Statistics Table ............ 69
3.2.7 SetTeamNames Table ........... 70
3.2.8 SetViewProjects Table .......... 71
3.2.9 SelectTeam Table ............ 72
3.2.10 MessageBoard Table .......... 73
3.2.11 Project Table .............. 74
3.2.12 Note Table ................. 75
3.2.13 Hours Table ............... 77
3.2.14 Cost Table ................. 78
3.2.15 DefineThread and SetMethod Tables ... 79
3.2.16 ApproveThread Table .......... 81
3.2.17 ChatScript Table .......... 83
CHAPTER FOUR: SOFTWARE QUALITY ASSURANCE

4.1 Web Interface Testing ............................ 86
   4.1.1 Purpose .................................... 86
   4.1.2 Procedure .................................. 87
   4.1.3 Expected Results ............................ 87

4.2 Testing for Robustness ............................ 88
   4.2.1 Purpose .................................... 88
   4.2.2 Procedure .................................. 88
   4.2.3 Results ..................................... 88
      4.2.3.1 Invalid Login ......................... 88
      4.2.3.2 Invalid Email ......................... 89
      4.2.3.3 Blank Input ............................ 90
      4.2.3.4 Code as Input .......................... 92
      4.2.3.5 Invalid Hours and Budget Allocation .................................. 94
      4.2.3.6 Invalid Method Selection ............. 96
      4.2.3.7 Invalid Database Location .......... 97

CHAPTER FIVE: USERS MANUAL

5.1 Purpose ......................................... 99
5.2 Install Apache ................................... 99
5.3 Install My-Structured Query Language .......... 100
5.4 Install Oracle .................................. 100
5.5 Install Hypertext Pre-Processor ............... 101
5.5.1 Setup Hypertext Pre-Processor with Gentoo Operating System ................. 101

5.5.2 Setup Hypertext Pre-Processor with Red Hat 8.0 and Above ..................... 101

5.6 How to Extract the ReMoTe ........................................... 103

5.7 Setting Up the ReMoTe .............................................. 103

5.8 Using Concurrent Version System with ReMoTe .................................... 104

5.9 Source Code ......................................................... 105

CHAPTER SIX: CONCLUSIONS

6.1 Conclusions ......................................................... 106

6.2 Future Directions .................................................. 108

APPENDIX A: FILE STRUCTURE ....................................... 110

REFERENCES ............................................................. 121
LIST OF TABLES

Table 1. Definitions, Acronyms, and Abbreviations ........................................... 5
Table 2. User Table ................................................................. 65
Table 3. Uniform Resource Locator Table ......................................................... 66
Table 4. Threads Table ............................................................... 67
Table 5. ThreadBlob Table ........................................................................... 68
Table 6. Settings Table ................................................................................. 69
Table 7. Statistics Table ............................................................................... 70
Table 8. SetTeamNames Table ....................................................................... 71
Table 9. SetViewProjects Table ..................................................................... 72
Table 10. SelectTeam Table ......................................................................... 73
Table 11. MessageBoard Table ....................................................................... 74
Table 12. Project Table ................................................................................. 75
Table 13. Note Table ..................................................................................... 76
Table 14. Hours Table .................................................................................... 77
Table 15. Cost Table ....................................................................................... 78
Table 16. SetMethod Table ............................................................................ 81
Table 18. ApproveThread Table ..................................................................... 82
Table 19. ChatScript Table ............................................................................ 84
Table 20. ChatUpload Table ......................................................................... 85
Table 21. ChatUploadBlob Table .................................................................. 85
Table 22. Results of the Web Test Cases ......................................................... 87
Table 23. Comparison of Microsoft Project and ReMoTe .................................... 107
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Deployment Diagram</td>
<td>8</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Use-Case Diagram</td>
<td>13</td>
</tr>
<tr>
<td>Figure 3</td>
<td>Main Logo</td>
<td>14</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Login Interface</td>
<td>15</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Administrator Welcome Screen</td>
<td>16</td>
</tr>
<tr>
<td>Figure 6</td>
<td>Administrator Project Name Settings</td>
<td>17</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Administrator User List</td>
<td>18</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Administrator Editing a User</td>
<td>19</td>
</tr>
<tr>
<td>Figure 9</td>
<td>Administrator Deleting User</td>
<td>20</td>
</tr>
<tr>
<td>Figure 10</td>
<td>View Only Person Welcome Screen</td>
<td>21</td>
</tr>
<tr>
<td>Figure 11</td>
<td>View Only Person Setting Projects to View</td>
<td>22</td>
</tr>
<tr>
<td>Figure 12</td>
<td>View Only Person Viewing Multiple Projects</td>
<td>23</td>
</tr>
<tr>
<td>Figure 13</td>
<td>Project Manager Welcome Screen</td>
<td>24</td>
</tr>
<tr>
<td>Figure 14</td>
<td>Project Manager Settings Page</td>
<td>25</td>
</tr>
<tr>
<td>Figure 15</td>
<td>Set Life-Cycle Model Page</td>
<td>26</td>
</tr>
<tr>
<td>Figure 16</td>
<td>Set Life-Cycle Model Step 1</td>
<td>27</td>
</tr>
<tr>
<td>Figure 17</td>
<td>Set Life-Cycle Model Step 2</td>
<td>28</td>
</tr>
<tr>
<td>Figure 18</td>
<td>Select Teammates Screen</td>
<td>29</td>
</tr>
<tr>
<td>Figure 19</td>
<td>Set Budget Screen</td>
<td>30</td>
</tr>
<tr>
<td>Figure 20</td>
<td>Set Man-Hours Screen</td>
<td>31</td>
</tr>
<tr>
<td>Figure 21</td>
<td>Set Team Names Screen</td>
<td>32</td>
</tr>
<tr>
<td>Figure 22</td>
<td>Manage Thread Screen</td>
<td>33</td>
</tr>
<tr>
<td>Figure 23</td>
<td>Approve/Deny Phase Screen</td>
<td>34</td>
</tr>
</tbody>
</table>
Figure 66. Invalid Hours Allocation ....................... 95
Figure 67. Invalid Budget Allocation ....................... 96
Figure 68. Invalid Method Selection ....................... 97
Figure 69. Invalid Database Location ....................... 98
CHAPTER ONE
INTRODUCTION

1.1 History

The origin of the Recursively Estimating Multi-threaded Observation Technology Enterprise (ReMoTe) is the Recursive Multi-Threaded (RMT) software life-cycle model and process by Scott Simon [7], which was created as a thesis. The RMT eventually evolved into a working prototype by Sunny Lin [6]. The RMT supported software process that is based on object-oriented approach. It is a software process management tool designed to address common problems in software companies such as: project planning, team hierarchy, software artifact organization, and repeatability. The RMT originally supported the following:

1. Team Organization
2. Progress Percentage
3. Software Artifact Organization
4. Web-based output (in Java Server Pages)
With this philosophy, it leads to the development of the ReMoTe Tool. Using the object-oriented approach of creating the current tool, ReMoTe can support any software lifecycle model. ReMoTe can help manage and control the software process over the Web. With Summer Xia [9] and Yi-Chen Kuo [4] the ReMoTe can now also support multiple databases.

1.2 Purpose of the Project

The purpose of the project is to provide the Department of Computer Science at California State University, San Bernardino with a software project management tool that will help software companies in their software development. The ReMoTe will assist software engineering teams with defining their scheduled delivery dates, life-cycle definitions, team hierarchy, and communication. An entire project organization will make use of the tool. The organization may consist of: system administrator, project manager, team leaders, software engineers, QA engineers, and architects. The new functions of ReMoTe will include the following:

1. Tracking the progress of software development based on any of the company's software life-cycle models.
2. Storage and retrieval of all software artifacts from software requirements specification to source codes.
3. Assignment and management of tasks of all personnel involved in the software project.
4. Traceability of software requirements specs from design to coding.
5. Link to common tools that the company uses, such as debugging, messaging, and chat rooms.
6. Support of multiple component software projects being developed in geographically different locations.

1.3 Improvements to Recursive Multi-Thread

The ReMoTe tool will recursively retrieve and calculate information of the ReMoTe structure in real time over the Web, unlike the RMT where users supplied percentages. When one member of the project adds or removes an artifact from his or her thread, everyone on the project will know who is behind on the project based on the progress percentage.

The effectiveness of ReMoTe comes from its intrinsic recursive design, in that all aspects of a project are handled in the same way, programmatically, while still allowing each team its own autonomy of designing their own thread designs. In other words, while ReMoTe is inherently
recursive, implying that all phases must be very nearly identical, this is not the case with ReMoTe.

ReMoTe will allow each level of development to utilize the most effective scheme for their part of the project, while remaining consistent with the overall software-processing paradigm.

Another improvement to the ReMoTe is the ability to track the timeline of the actual progress of the project. Managers will be able to check if the teams will be able to complete a phase on time or not with the ReMoTe’s Gantt chart generator.

The ReMoTe can also allow teams to use different life-cycle models to accommodate tasks that are not concerned with software development, such as documentation, systems testing, etc.

Also, other improvements in the current version of the ReMoTe will allow team to communicate in various formats, such as a message board and chat rooms. The chat rooms also allow teammates from different parts of the world to share files, such as UML diagrams, video, documents, etc. to effectively communicate ideas and information successfully over the web.
CHAPTER TWO
SOFTWARE REQUIREMENTS SPECIFICATION

2.1 Scope

The ReMoTe user interface will provide an easy-to-use interface that will allow team members to define and setup their project and will be able to manage it from the Web by an online browser, such as Internet Explorer, Firefox, etc. The ReMoTe will help managers in predicting the deadline for a project based on the activity on ReMoTe's tracking system.

2.1.2 Definitions, Acronyms, and Abbreviations

See Table 1 for list of the Definitions, acronyms, and abbreviations mentioned in the Software Requirements Specification (SRS).

Table 1. Definitions, Acronyms, and Abbreviations

<table>
<thead>
<tr>
<th>CSCI</th>
<th>Computer Science</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSS</td>
<td>Short for Cascading Style Sheets, a new feature being added to HTML that gives both web-site developers and users more control over how web-pages are displayed. With CSS, designers and users can create style sheets that define how different elements, such as headers and links, appear. These style sheets can then be applied to any Web page.</td>
</tr>
</tbody>
</table>
The term cascading derives from the fact that multiple style sheets can be applied to the same Web page. CSS was developed by the W3C [7].

<table>
<thead>
<tr>
<th>ER</th>
<th>Entity Relationship</th>
</tr>
</thead>
<tbody>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HTML</td>
<td>Hyper Text Markup Language</td>
</tr>
<tr>
<td>IE</td>
<td>Internet Explorer</td>
</tr>
<tr>
<td>Iteration</td>
<td>Iteration is the repetition of a process. It describes a specific form of repetition with a mutable state. It also can be considered as a different version of a project [6]</td>
</tr>
<tr>
<td>Javascript</td>
<td>A scripting language developed by Netscape to enable Web authors to design interactive sites. JavaScript can interact with HTML source code, enabling Web authors to spice up their sites with dynamic content. JavaScript is endorsed by a number of software companies and is an open language that anyone can use without purchasing a license [7]</td>
</tr>
<tr>
<td>Linux</td>
<td>A multi user UNIX operating system</td>
</tr>
<tr>
<td>mySQL</td>
<td>My-Structured Query Language</td>
</tr>
<tr>
<td>MS</td>
<td>Microsoft</td>
</tr>
<tr>
<td>OO</td>
<td>Object Oriented</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>PHP</td>
<td>Hypertext Preprocessor</td>
</tr>
<tr>
<td>RMT</td>
<td>Recursive Multi-threaded Tool</td>
</tr>
<tr>
<td>SRS</td>
<td>Software Requirements Specification</td>
</tr>
<tr>
<td>SQL</td>
<td>Structured Query Language</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>Abbreviation of Transmission Control Protocol. TCP is one of the main protocols in TCP networks. Whereas the IP protocol deals only with packets, TCP enables two hosts to establish a connection and exchange streams of data. TCP guarantees delivery of data and also guarantees that packets will be delivered in the same order in which they were sent [7]</td>
</tr>
<tr>
<td>Thread</td>
<td>A Thread contains a person's individual artifact(s) for a given iteration for a prototype in their project [6]</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
</tbody>
</table>
2.1.3 Overview

The rest of this document will have two major sections and Appendixes and will be ended with an index for the Software Requirement Specifications (SRS) documents [3]. The second section will contain all the product outlines, perspective, function, interface, and user characteristics. The third section will include the some specific requirements (e.g. hardware requirements).

2.2. Overall Description

2.2.1 Product Perspective

The ReMoTe project will provide users a detailed look at the progress of a software project by giving team members a progress report percentage of: individual member, team(s), and the entire project. The project will also supply a Gantt chart, statistical bar chart, and provide the artifacts supplied to the ReMoTe to all users in the project.

2.2.2 System Interfaces

The interface for this software is a web interface. You will be able to use the software via any browser (Mozilla, Firefox, Internet Explorer, etc.) The ReMoTe can have multiple instances using different databases, which also allows managers to view multiple projects at once.
Each server can have Oracle, mySQL, or Microsoft Access as its database and the client can view all three servers at the same time (see Figure 1).

Figure 1. Deployment Diagram.
2.2.3 User Interfaces

Since there are different levels of users, there must be a different interface for each of them. Any of the user interfaces will include a login interface page, which will allow them to input their username and password to login. Also all members (except Administrator) can use the communication tools: message board, chat room, team notes/tasks. The list of users and tasks are the following:

a. System Administrator:
   - Approve/Deny users.
   - Edits users.
   - Create Project Names
   - Set special types of users:
     i. CEO (View Only)
     ii. Project Manager

b. View Only Person:
   - Set projects to view.
   - View multiple projects at once (progress).

c. Project Manager:
   - Set welcome message.
   - Set Bugzilla Location (optional)
   - Set number of iterations in project.
- Define life-cycle model.
- Approve/Deny members’ threads.
- Set team names.
- Set budget.
- Set man-hours.
- Manage Thread:
  i. Adding/Deleting software artifacts

d. Team Leader:

- Define life-cycle model.
- Approve/Deny members’ threads.
- Set team names.
- Set budget.
- Set man-hours.
- Manage Thread:
  i. Adding/Deleting software artifacts

e. Employee

- Manage Thread:
  i. Adding/Deleting software artifacts

2.2.4 Hardware Interfaces

No hardware interface.

2.2.5 Software Interfaces

The project software interface will be viewed over the web. It can run on any web browser for Windows, Linux,
or Mac OS. The language and applications used in writing this interface are Javascript, Flash, PHP 4.3.1, and CSS. The server will run Gentoo Linux.

2.2.5.1 Communication Interface. ReMoTe uses Hypertext Pre-Processor (PHP) and the operating system to manage the communication between the client and the server. The mySQL and Oracle databases will use PHP's built in function to handle communication between PHP and the their database. Microsoft Access on the other hand will use ODBC to communicate between PHP and the Microsoft Access database.

2.2.5.2 Memory Constraints. For ReMoTe server with mySQL/Access database:

- 256 MB or greater memory size.
- PII 500 or greater.

For ReMoTe server with Oracle 9i database:

- 512 MB or greater memory size.
- PIII 1Ghz
- 20GB disk space.

Client:

- 128 MB or greater memory size.

2.2.5.3 Operations. The ReMoTe will operate 24/7. Backups can be done of the database once a month.
Maintenance will be done on call, and mostly done remotely.

2.2.5.4 Site Adaptation Requirements. The ReMoTe will run from the Department of Computer Science at California State University, San Bernardino.

2.2.6 Product Functions

The use case diagram below depicts the typical users of the ReMoTe: system administrator, view only person, Project Manager, team leader, and employee. This also contains the actions that the project is expected to perform for the users (see Figure 2).
2.2.7 User Characteristics

The projected users of the ReMoTe will include software engineers, including the CEO (view only person). These individuals should be familiar with computer
operations in general and should not need additional basic instructions to use this product. All instructions and/or guidance can be provided using the ReMoTe’s help menu.

2.2.8 Constraints

The ReMoTe will be fully functional and will be demonstrated at the end of the Winter 2006 quarter.

2.2.9 Assumptions and Dependencies

There are no assumptions and dependencies.

2.2.10 Apportioning of Requirements

There are no apportioning of requirements.

2.3 Specific Requirements

2.3.1 External Interface

The main logo of the ReMoTe will appear on the top left corner of the page (see Figure 3).

Figure 3. Main Logo
2.3.2 User Interface

2.3.2.1 Login Interface. When the user visits the ReMoTe site for the first time, they will have to login to be able to use the tool. If they do not have a username/password, they must register so that the admin will approve the account to allow the users in (see Figure 4).

![Login Interface Diagram]

Figure 4. Login Interface

2.3.2.2 Administrator Interface. Once the administrator has successfully logged into the tool, they will be able to select a list of options on the left side of the screen (see Figure 5).
2.3.2.2.1 Set Project Names. In order for users to register in the ReMoTe system, the administrator must first supply project names to the system (see Figure 6).
2.3.2.2.2 Edit Users. The administrator can modify the users information supplied to the ReMoTe. In order to do this they must select a user from the list (see Figures 7 and 8).
### Figure 7. Administrator User List

<table>
<thead>
<tr>
<th>Name</th>
<th>Username</th>
<th>Type</th>
<th>Project Name</th>
<th>Approved</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guest, Ima A.</td>
<td>guest</td>
<td>P</td>
<td>GUEST</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>AlBajo, Alva I.</td>
<td>alva</td>
<td>N</td>
<td>TEAM1</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Brauer, Joel J.</td>
<td>jbrauer</td>
<td>P</td>
<td>TEAM1</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Concepcion, Arturo I.</td>
<td>concep</td>
<td>V</td>
<td>TEAM1</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>DSouza, Sammy R.</td>
<td>sammy</td>
<td>N</td>
<td>TEAM1</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Getahun, Abenet m.</td>
<td>agetahun</td>
<td>N</td>
<td>TEAM1</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Hung, Cheng-Yu.</td>
<td>justin</td>
<td>N</td>
<td>TEAM1</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Wiser-Orozco, Natalie A.</td>
<td>lasrcista</td>
<td>N</td>
<td>TEAM1</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Akbulak, Aiper</td>
<td>eagle</td>
<td>N</td>
<td>TEAM2</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Coulier, Robert J.</td>
<td>rcoulier</td>
<td>N</td>
<td>TEAM2</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Dang, Minhchau</td>
<td>mdang</td>
<td>P</td>
<td>TEAM2</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Shi, Wei</td>
<td>vshi</td>
<td>N</td>
<td>TEAM2</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Shihadeh, Fadi E.</td>
<td>fsihade</td>
<td>N</td>
<td>TEAM2</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Chung, Young S.</td>
<td>ychung</td>
<td>N</td>
<td>TEAM3</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>DeMelo, Darrion T.</td>
<td>ddsmelo</td>
<td>P</td>
<td>TEAM3</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Fan, Yao-Lung</td>
<td>yao</td>
<td>N</td>
<td>TEAM3</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Tafas, Jihad J.</td>
<td>jtafas</td>
<td>N</td>
<td>TEAM3</td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Zuriekat, Fares N.</td>
<td>Fares</td>
<td>N</td>
<td>TEAM3</td>
<td>Yes</td>
<td>Edit</td>
</tr>
</tbody>
</table>
Figure 8. Administrator Editing a User

2.3.2.2.3 Delete Users. The administrator can also delete a user (see Figure 9).
Figure 9. Administrator Deleting User

2.3.2.3 View Only Interface. Once the view only person has successfully logged into the tool, they will be able to select a list of options on the left side of the screen (see Figure 10).
2.3.2.3.1 View Only Person Set Projects to View.

After login, the view only person can select which projects to view (see Figure 11).
2.3.2.3.2 View Only Person Viewing Multiple Projects. After the view only person supplies the projects to view, they can view the entire project and get the overall project, the teams, progress, and individual member's progress. This will help in determining which team is holding up the progress of the project (see Figure 12).
2.3.2.4 Project Manager/Team Leader. Once the project manager/Team Leader has successfully logged into the tool, they will be able to select a list of options on the right side of the screen (see Figure 13).

Figure 12. View Only Person Viewing Multiple Projects
Figure 13. Project Manager Welcome Screen

2.3.2.4.1 Set Project Settings. Once the project manager has successfully logged into the tool, they will be able to manage the project such as Bugzilla location, set the number of iterations in a project, and also set the welcome message for other users to see (see Figure 14).
2.3.2.4.2 Define Life-Cycle Model. Threads in the ReMoTe can be individually defined to allow teammates to accommodate tasks that are not concerned with software development, such as documentation, system testing, etc. Thus, the project manager, including the team leaders, can supply the life-cycle model for the users threads that are one level below them (see Figures 15, 16, and 17). The managers must go through the following to create a life-cycle:
Step 1. Set the life-cycle name and the number of phases for the life-cycle.

Step 2. Define the life-cycle's phase names and timeline for each phase.

Figure 15. Set Life-Cycle Model Page
Figure 16. Set Life-Cycle Model Step 1
2.3.2.4.3 Select Teammates. Once the life-cycle model is defined, the user can supply their team, they will be given a list of members in the ReMoTe, the users responsibility to select the members of the persons team, which consists of people one level below the user, the user can place users into different life-cycle models (see Figure 18).
2.3.2.4.4 Set Budget. An option for the project manager and/or the team leader(s), they can supply the budget for the entire project and distribute it to the respective teams (see Figure 19).
Figure 19. Set Budget Screen

2.3.2.4.5 Set Man-Hours. Another option for the project manager and/or the team leader(s), they can supply the man-hours for the entire project and distribute it to the respective teams (see Figure 20).
2.3.2.4.6 Set Team Names. Another option for the project manager and/or the team leader(s) is that they can supply the team-names to their respective teammates (see Figure 21).
2.3.2.5 Manage Thread Interface. The Project Manager, team leaders, and employees are given the task of supplying the ReMoTe the software artifacts for managers to approve and/or deny. The artifacts are uploaded to the ReMoTe and the user specifies what phase the files should be placed into. There is no limitation of file types that can be uploaded to the ReMoTe (see Figure 22).
2.3.2.5.1 Approve/Deny Thread. The project manager and team leaders have the responsibility to approve each phase during the process of the project. They can review the artifacts and send a comment on why the file was approved/denied (see Figure 23).
2.3.2.6 User Communication Interface. All users in the ReMoTe will be able to use a wide range of communication tools to help members, local and abroad, to be able to be in contact with other members in one easy-to-use tool.

2.3.2.6.1 Message Board. All members can use the message board to leave messages at their own pace in the ReMoTe’s message board system (see Figures 24 and 25).
Figure 24. Message Board List

Figure 25. Message Board Thread
2.3.2.6.2 Chat Room. All members can use the chat room feature to communicate in real-time to other users and also be able to share documents live. Users can also invite other users outside the ReMoTe to the chat room (see Figures 26 and 27).

Figure 26. Chat Room Sharing a Unified Modeling Language Diagram
2.3.2.6.3 Set Task/Note. All members can use the task/note feature to alert users when they log in of any tasks of notes for members to do (see Figures 28 and 39).
When you get a chance, please do research on PHP and JavaScript so you will have a better understanding of the syntax and structure of the program.

Thais,

Description:

- Study PHP/JavaScript

Create/Write Tasks

Subject:
2.3.2.7 Viewing Project Interface. All users in the ReMoTe will be able view the progress of the entire project and an individual’s progress and overall progress (see Figures 30 and 31). For individual progress, you can notice the uses phases/tasks they supplied in their lifecycle (see color-key in Figure 30). You can notice which phases the individual has improved in their progress. Also, you can notice if the user later in the process of the project decides to change the number of phases, the chart will note any of the changes.
For the overall progress, you can view the progress of the project based on the phases that were defined by the project manager.

Figure 30. Individual Progress Chart
2.3.2.7.1 Gantt Chart. Once the users has been given a life-cycle model to use for their thread. The ReMoTe keeps track of the users real activity. The ReMoTe's Gantt chart generator shows the defined project plan and compares it to the actual project plan, if any of

Figure 31. Overall Progress Chart
the phases has not been approved; the ReMoTe will delay the expected end date until it has been completed (see figure 32).

Figure 32. Gantt Chart Generator
2.3.3 Functions

All additional functions of ReMoTe tool are described in section 2.

2.3.4 Performance Requirements

Code will be optimized for functionality rather than speed of operation.

2.3.5 Logical Database Requirements

We will be using mySQL, Microsoft Access, and Oracle database; the ports for each of the databases must be open on the server to allow access for users.

2.3.6 Design Constraints

2.3.6.1 Standards Compliance. The ReMoTe tool will comply with PHP coding standards.

2.3.7 Software System Attributes

2.3.7.1 Reliability. The servers running the ReMoTe will be completely functional.

2.3.7.2 Availability. The ReMoTe tool will require a computer with 128 megabytes of ram and World Wide Web access available 24/7.

2.3.7.3 Security. The ReMoTe tool will require a computer with 128 megabytes of ram and World Wide Web access available 24/7.

2.3.7.4 Maintainability. The code will be well documented and most modules will be reusable.
2.3.7.5 Portability. The code used in this program will be completely portable, allowing the code to go from one server to another as long they who comply with the hardware and software requirements detailed in this document.

2.3.8 Organizing the Specific Requirements

2.3.8.1 System Mode. Single mode of operation (normal mode).

2.3.8.2 User Class. There are three users of the ReMoTe tool: system administrator, CEO, Project Manager, Team Leader, and Employee.

2.3.8.3 Objects. See the Use-Case Diagram in Figure 2.

2.3.8.4 Feature. This software contains no external features.

2.3.8.5 Stimulus. This software supports no external stimuli.

2.3.8.6 Response. All responses to user input are described in section 2.

2.3.8.7 Functional Hierarchy. Section 2.1.2 describes the control flow of the ReMoTe tool.

2.4 Additional Comments

No additional comments.
CHAPTER THREE

SYSTEM AND SOFTWARE DESIGN

3.1 MVC Architecture Design

In both [9] and in this project, the ReMoTe uses the Model View Controller (MVC) architecture. See Figure 33 for ReMoTe’s MVC architecture design. The MVC design pattern provides a way to help organize information to display to the user. The MVC consists of [9,8]:

i. Model: The object that contains data i.e. domain logic.

ii. View: Gives the presentation of the application based on what the model renders.

iii. Controller: An event provided by the user that causes change of the view.
Controller
-- handles input events from user.

Model
-- domain logic

View
Responsibilities
-- displays information about the model

Figure 33. Model View Controller Architecture Design
3.1.1 Controller

When a user clicks on a hyperlink on the ReMoTe, the user is sending an operation to the controller class. The controller class then sends the user to their respective action according to what they supplied. For example, if the user clicked on the "Set Hours" link, the value "setHours" is sent to the controller, the controller identifies the operation and sends the information to the "setHoursAction" class, which then renders (models) the
input interface in the user’s browser for setting the projects man-hours. Therefore, for every action a user can perform on the ReMoTe, there is an action that corresponds to it. There are action files for both the administrator (see Figure 35) and for the user (see Figure 36).

3.1.2 Administrator Action Files

For the administrator, there are 13 action files that correspond to each action an administrator can take in the ReMoTe. This is considered the "model" part of the MVC architecture. The action file and their actions include:

3.1.2.1 AcceptUserAction. Approves the user into the ReMoTe system.

3.1.2.2 CreateProjectAction. Creates the project name for identification for the user’s registration.

3.1.2.3 DeleteMessageAction. Deletes threads in ReMoTe’s message board system.

3.1.2.4 DeleteProjectAction. Deletes project names in the system.

3.1.2.5 DeleteUserAction. Delete users in the ReMoTe system.

3.1.2.6 EditMessageAction. Edits a message in the ReMoTe message board.

3.1.2.7 EditUserAction. Edits a user’s contact information and/or username or password in the system.
3.1.2.8 **EditUserPageAction.** Displays the list of users in the system to allow the administrator to choose to edit.

3.1.2.9 **LoginAction.** Manages the login for the administration system.

3.1.2.10 **RegistrationAction.** Submits the newly updated user’s information to the database.

3.1.2.11 **SetProjectManagerAction.** Sets the role of a user to project manager.

3.1.2.12 **SetViewOnlyAction.** Sets the user’s role to viewer, which allows them to handle viewing multiple projects.
3.1.3 User Action Files

For the users of the ReMoTe system, there are 32 action files that correspond to each action an administrator can take in the ReMoTe. There are also 5 sub-action files that pertain to the thread output of the ReMoTe. The action file and their actions include:

3.1.3.1 AddThreadAction. Uploads the user’s software artifacts to the ReMoTe.
3.1.3.2 **ApproveThreadAction.** Allows team leaders/managers to approve the phases of a user’s thread in the project.

3.1.3.3 **CostAction.** Submits the money denomination (cost) allotted for each individual team in the project.

3.1.3.4 **CostAnalysisAction.** Returns to the user the list of monetary values allotted for each team in the project for review.

3.1.3.5 **CreateThreadAction.** Returns to the user defined phases for the user to supply software artifacts to.

3.1.3.6 **DefineThreadAction.** Allows the team leader/manager to set the number of phases and name of the life-cycle model.

3.1.3.7 **DeleteFileAction.** Allows the user to delete software artifacts in their thread.

3.1.3.8 **DeleteMethodAction.** Deletes a life-cycle method supplied by the user.

3.1.3.9 **DeleteNoteAction.** Deletes a note/task defined by the user.

3.1.3.10 **EditMethodAction.** Allows the user to modify the values in their supplied life-cycle model.

3.1.3.11 **EditNoteAction.** Allows the user to modify the values in their supplied note/task.
3.1.3.12 **EditUserAction.** Allows the user to modify their contact information and/or their username and password.

3.1.3.13 **ForgotPasswordAction.** Returns the user's password they have forgotten. Note: the user must supply a valid email and username to the system in order to retrieve the lost password.

3.1.3.14 **GanttChartAction.** Returns the user’s timeline for the phases of their project in a Gantt chart format.

3.1.3.15 **HoursAction.** Submits to the database the man-hours allotted for each team in the project.

3.1.3.16 **LoginAction.** Validates the username and password to log into the system.

3.1.3.17 **MessageBoardAction.** Submits to the database a new thread to ReMoTe’s message board system.

3.1.3.18 **MessageCheckAction.** Validates if to show the user receives an animation of a new message for them.

3.1.3.19 **NoteAction.** Submits the defined notes/tasks for the team to utilize.

3.1.3.20 **NoteListAction.** Displays the entire list of notes/task defined for the user’s team for review.
3.1.3.21 OverridePhaseNamesAction. Allows the user to modify the names of the phases defined in the user-defined life-cycle model.

3.1.3.22 RegistrationAction. Submits the users contact information, username and password to the system.

3.1.3.23 SelectTeamAction. Allows the user to supply the users the team leader/manager is responsible for.

3.1.3.24 SetDatesAction. Supplies the database the start date and number of days for a given life-cycle model.

3.1.3.25 SetMethodAction. Supplies the name of a given life-cycle model.

3.1.3.26 SetNumberPhasesAction. Supplies the number of phases for a given life-cycle model.

3.1.3.27 SetPictureAction. Allows the user to upload a photo to the system.

3.1.3.28 SetProjectWeights. Allows the users with the “viewer” roles to supply weights to team that maybe more important than others.

3.1.3.29 SettingsAction. Allows the project manager to supply the Bugzilla location, number of iterations in the project, and an introduction message for users to see when they log into the system.
3.1.3.30 ViewMessageAction. Displays the user an individual message in the message board system.

3.1.3.31 ViewMessagesAction. Displays the list of messages in the project message board.

3.1.3.32 ViewNotesAction. Displays the list of notes for modification/deletion.

3.1.3.33 ShowFileAction. From the thread output section, when a user views a members software artifact. It displays the file in the browser in the form of pure text for text files or displays images.

3.1.3.34 ShowItem. Allows the user to download the file or view from a browser.

3.1.3.35 ShowTeamListAction. Displays the team hierarchy of the project in the browser.

3.1.3.36 StatisticsAction. From the thread output section, it displays the progress of the project in a bar chart format.
3.1.4 Interface Files

All of the HTML files are stored in an HTML folder with the extension *.inc. This helps the programmer determine the difference between program purely coded in PHP (ends with *.php) or is an HTML file (* .inc). All
HTML files are solely meant for the interface to render for the user to view (see Figures 37 and 38 for both the administrator and user interface files).

For every action a user sends, the controller determines the layout the user would receive based on the action of the user. These layouts allow the programmer for adjustability and reusability of code when needed.

Inside every template file, there is basically two views the user may receive based on their actions: first-time and submission. Template files end with the extension *.tpl.

The first-time view is when a user visits the page for the first time, in order for the ReMoTe to recognize this there is NULL values sent to the action class, in PHP, it is in the form of $_POST or $_GET which is an array contained in a form or in a URL query respectively. The action class would recognize the NULL values and return the user with an input page. The other type of view is when the user submits the form to the controller.

In submission view, the values in their form are sent to the action, the action then processes the data. If the data is valid it is either sent to a database or the user is redirected to another page. If the data is invalid or missing, the user is then halted of process and are
redirect to a page determined by the controller and receives and response of their invalid data. For example, if a user was at the registration page and supply their user information, but forgot to enter their e-mail address on submission, the controller will send the user back to the registration page and are informed of their blank value for their e-mail in the form of a warning in bold red font in the top-center of the page.

![Diagram](image.png)

Figure 37. Administration Interface Package
3.2 Database Design

In the ReMoTe, the database is crucial for storing software artifacts to help in both data retrieval and statistical analysis of the progress of the project. The
ReMoTe is able to handle the communication between three different databases: mySQL, Microsoft Access, and Oracle. The administrator can specify which database to use in the ReMoTe’s setup file, see Figure 39.

```php
<?php

// put username to get into database
$user = "root";

// put password to get into database
$pwd = "myPassword";

// Title of database
$db = "myDatabase";

// database type
// 1 == mySQL
// 2 == Microsoft Access
// 3 == Oracle
// 4 == Experimental
$db_type = "1";

// host of database
$host = "localhost";

// company name (Please use text only)
$companyname = "Masters Project";

// Set username, and password for administrator to create project managers
$admin_name="Admin";
$admin_pass="Admin";

People involved:
Darrius DeNoelo
David Hollingsworth
Chaz Lee
Norman Loenendi

Figure 39. Setup File
```
All of the database functionality of the ReMoTe are stored in Database Access Objects (DAO). The DAO files call on a ConnectDAO class, which handles all of the queries to the different databases (see Figure 40 and 41 for ER Diagram).
Figure 40. Database Listing and Architecture
Figure 41. Entity Relationship Diagram of the Database
3.2.1 User Table

The user table contains all of the user's contact information and picture (if supplied). It also has the role of the user: viewer, project manager, none. Each role has different functions that will be available to them. See Figure 42 and Table 2.

The viewer role only can view multiple projects, Chief Executive Officers (CEO) typically use this. Project managers define the project and the core elements of the ReMoTe, i.e. cost, man-hours, settings, etc.
Figure 42. Entity Relationship Diagram of User
Table 2. User Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>mysql</td>
<td>Access</td>
<td>Oracle</td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>username</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>password</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>firstName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>lastName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>middleInitial</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>phone</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>email</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>role</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>valid</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>address</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>pictureName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>pictureType</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>pictureSize</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>pictureContent</td>
<td>blob</td>
<td>memo</td>
<td>blob</td>
</tr>
</tbody>
</table>

3.2.2 Uniform Resource Locator Table

The Uniform Resource Locator (URL) table is responsible for handling the URL location of the ReMoTe, this helps with the database of projects and file retrieval of artifacts that are on different servers. See Figure 43 and Table 3.
3.2.3 Threads and ThreadBlob Table

The thread table is responsible for handling the user’s software artifacts. Each artifact is stored in the database in the form of a Binary Large Object (BLOB) in the ThreadBlob table. See Figure 44 and Tables 4 and 5.
Figure 44. Entity Relationship Diagram of Threads and ThreadBlob

Table 4. Threads Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td></td>
<td></td>
</tr>
<tr>
<td>companyName</td>
<td>text</td>
<td></td>
<td></td>
</tr>
<tr>
<td>iteration</td>
<td>int</td>
<td></td>
<td></td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fileName</td>
<td>text</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fileType</td>
<td>text</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fileSize</td>
<td>int</td>
<td></td>
<td></td>
</tr>
<tr>
<td>versionNumber</td>
<td>double</td>
<td></td>
<td></td>
</tr>
<tr>
<td>userId</td>
<td>int</td>
<td></td>
<td></td>
</tr>
<tr>
<td>phaseId</td>
<td>int</td>
<td></td>
<td></td>
</tr>
<tr>
<td>location</td>
<td>text</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

mysql Access Oracle

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>PRI</td>
<td></td>
</tr>
<tr>
<td>companyName</td>
<td>text</td>
<td></td>
<td></td>
</tr>
<tr>
<td>iteration</td>
<td>int</td>
<td>MUL</td>
<td></td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>MUL</td>
<td></td>
</tr>
<tr>
<td>fileName</td>
<td>text</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fileType</td>
<td>text</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fileSize</td>
<td>int</td>
<td></td>
<td></td>
</tr>
<tr>
<td>versionNumber</td>
<td>double</td>
<td></td>
<td></td>
</tr>
<tr>
<td>userId</td>
<td>int</td>
<td>MUL</td>
<td></td>
</tr>
<tr>
<td>phaseId</td>
<td>int</td>
<td></td>
<td></td>
</tr>
<tr>
<td>location</td>
<td>text</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 5. ThreadBlob Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mysql</td>
<td>Access</td>
<td>Oracle</td>
</tr>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>content</td>
<td>blob</td>
<td>memo</td>
<td>blob</td>
</tr>
<tr>
<td>fileId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
</tbody>
</table>

3.2.4 Settings Table

Each project manager in the ReMoTe is responsible for providing the number of iterations of the project, Bugzilla location (if needed), and an introduction message to all members of the project (see Figure 45 and Table 6).

![Figure 45. Entity Relationship Diagram of Settings](image-url)
### Table 6. Settings Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>companyName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>bugzilla</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>message</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>numberIterations</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
</tbody>
</table>

#### 3.2.6 Statistics Table

The statistics table contains all of the progress information for every individual user, the user’s team and the overall project (see Figure 46 and Table 7).

![Entity Relationship Diagram of Statistics](image)

Figure 46. Entity Relationship Diagram of Statistics
### Table 7. Statistics Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>mySQL</td>
<td>Access</td>
<td>Oracle</td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number PRI</td>
</tr>
<tr>
<td>iteration</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>companyName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)MUL</td>
</tr>
<tr>
<td>percentage</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>userId</td>
<td>text</td>
<td>text</td>
<td>varchar(240)MUL</td>
</tr>
<tr>
<td>date</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>totalCorrect</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>numberPhases</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
</tbody>
</table>

### 3.2.7 SetTeamNames Table

The SetTeamNames table allows the project manager/team leader to rename their team's one level below them (see Figure 47 and Table 8).

![Figure 47. Entity Relationship Diagram of SetTeamNames](image)

Figure 47. Entity Relationship Diagram of SetTeamNames

70
Table 8. SetTeamNames Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mySQL</td>
<td>Access</td>
<td>Oracle</td>
</tr>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>companyName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>iteration</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>teamId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>teamName</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
</tbody>
</table>

3.2.8 SetViewProjects Table

The SetViewProjects table allows the users that have the viewer role to handle which projects to view by providing the ReMoTe the database locations and the database type (see Figure 48 and Table 9).

![Entity Relationship Diagram of SetViewProjects](image-url)

Figure 48. Entity Relationship Diagram of SetViewProjects
Table 9. SetViewProjects Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mySQL</td>
<td>Access</td>
<td>Oracle</td>
</tr>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>location</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>companyName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>dbType</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>username</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>password</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>personId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>databaseName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
</tbody>
</table>

3.2.9 SelectTeam Table

The SelectTeam table contains all of the users level in the ReMoTe and handles which user is responsible for and who is on the user's team (see Figure 49 and table 10).
Figure 49. Entity Relationship Diagram of SelectTeam

Table 10. SelectTeam Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>managerId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>subUserId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>iteration</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>methodId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
</tbody>
</table>

3.2.10 MessageBoard Table

The MessageBoard table has is the ReMoTe's message board system (see Figure 50 and Table 11).
Table 11. MessageBoard Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>userId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>date</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>broadcast</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>replyId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>message</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>subject</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
</tbody>
</table>

3.2.11 Project Table

The Project table contains the project name supplied by the administrator that helps with the user's registration (see Figure 51 and Table 12).
Figure 51. Entity Relationship Diagram of Project

Table 12. Project Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td></td>
<td>number</td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td></td>
<td>varchar(240)</td>
</tr>
</tbody>
</table>

3.2.12 Note Table

The Note table contains the notes/task a user would supply for their team (see Figure 52 and Table 13).
Figure 52. Entity Relationship Diagram of Note

Table 13. Note Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PRI</td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>iteration</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>note</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>title</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>userId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>dateCreated</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
</tbody>
</table>
3.2.13 Hours Table

The Hours table stores all of the man-hours required for a user’s team (see Figure 53 and Table 14).

Figure 53. Entity Relationship Diagram of Hours

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>MySQL</th>
<th>Access</th>
<th>Oracle</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
<td>number</td>
<td>PRI</td>
<td></td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
<td>MUL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>iteration</td>
<td>int</td>
<td>int</td>
<td>number</td>
<td>MUL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hours</td>
<td>int</td>
<td>int</td>
<td>number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>userId</td>
<td>int</td>
<td>int</td>
<td>number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>managerId</td>
<td>int</td>
<td>int</td>
<td>number</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.2.14 Cost Table

The Cost table stores all of the money denomination (cost) required for a user's team (see Figure 54 and Table 15).

Figure 54. Entity Relationship Diagram of Cost

Table 15. Cost Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td></td>
<td></td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>varchar(240)</td>
<td>MUL</td>
</tr>
<tr>
<td>iteration</td>
<td>int</td>
<td>number</td>
<td></td>
</tr>
<tr>
<td>cost</td>
<td>int</td>
<td>number</td>
<td>MUL</td>
</tr>
<tr>
<td>userId</td>
<td>int</td>
<td>number</td>
<td></td>
</tr>
<tr>
<td>managerId</td>
<td>int</td>
<td>number</td>
<td></td>
</tr>
</tbody>
</table>
3.2.15 DefineThread and SetMethod Tables

In order for the team to submit software artifacts to the ReMoTe, the user's respective team leader/manager must supply the team's life-cycle model/method for the team to use. Each team leader/manager can supply as many models needed for each member for a team (see Figure 55 and tables 16, 17).
Figure 55. Entity Relationship Diagram of

DefineThread and SetMethod
Table 16. SetMethod Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>name</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>iteration</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>userId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>numberPhases</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
</tbody>
</table>

Table 17. DefineThread Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>name</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>iteration</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>userId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>phaseNumber</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>phaseDays</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>startMonth</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>projectName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>methodId</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>startYear</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>startDay</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>phaseName</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
</tbody>
</table>

3.2.16 ApproveThread Table

The ApproveThread table stores the users approval state in the ReMoTe for each phase of the project defined by their team leader/manager. If the phase approved or
denied, the team leader must supply a reason for the state the user is in, i.e. invalid files, missing files, etc (see Figure 56 and Table 18).

Figure 56. Entity Relationship Diagram of ApproveThread

Table 18. ApproveThread Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>PRI</td>
<td></td>
</tr>
<tr>
<td>approve</td>
<td>int</td>
<td></td>
<td></td>
</tr>
<tr>
<td>iteration</td>
<td>int</td>
<td></td>
<td>MUL</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>PRI</td>
<td></td>
</tr>
<tr>
<td>approve</td>
<td>int</td>
<td></td>
<td></td>
</tr>
<tr>
<td>iteration</td>
<td>int</td>
<td></td>
<td>MUL</td>
</tr>
</tbody>
</table>
3.2.17 ChatScript Table

The ChatScript table contains all of the user’s text supplied to the chat room (see Figure 57 and Table 19).

![Entity Relationship Diagram of ChatScript]

Figure 57. Entity Relationship Diagram of ChatScript
Table 19. ChatScript Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mySQL</td>
<td>Access</td>
<td>Oracle</td>
</tr>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>theNick</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
<tr>
<td>projectName</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>chatTime</td>
<td>int</td>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>theText</td>
<td>text</td>
<td>text</td>
<td>varchar(240)</td>
</tr>
</tbody>
</table>

3.2.18 ChatUpload and ChatUploadBlob Tables

The ChatUpload and ChatUploadBlob tables contain the shareable items for the chat room. All data is stored in the form of a BLOB in the ChatUploadBlob while the ChatUpload table contains the data information, i.e. file name, file size, etc. (see Figure 58 and Table 20).

Figure 58. Entity Relationship Diagram of ChatUpload and ChatUploadBlob
### Table 20. ChatUpload Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>MySQL</th>
<th>Access</th>
<th>Oracle</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>int</td>
<td>number</td>
<td>PRI</td>
<td></td>
</tr>
<tr>
<td>fileSize</td>
<td>int</td>
<td>int</td>
<td>int</td>
<td>number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>projectName</td>
<td>int</td>
<td>int</td>
<td>int</td>
<td>number</td>
<td>MUL</td>
<td></td>
</tr>
<tr>
<td>chatTime</td>
<td>int</td>
<td>int</td>
<td>int</td>
<td>number</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 21. ChatUploadBlob Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>MySQL</th>
<th>Access</th>
<th>Oracle</th>
<th>Key</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>int</td>
<td>int</td>
<td>number</td>
<td>PRI</td>
<td></td>
</tr>
<tr>
<td>fileId</td>
<td>int</td>
<td>int</td>
<td>int</td>
<td>number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>content</td>
<td>blob</td>
<td>memo</td>
<td>blob</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4.1 Web Interface Testing

4.1.1 Purpose

For web testing, we must verify the following on each web page:

1. Stylesheet
   a. Verify that all interface design of the CSS will display the same for all browsers.
   b. Verify CSS validation using the World Wide Web Consortium (W3C) validation system.
      http://jigsaw.w3.org/

2. Javascript
   a. Verify that all JavaScript functions are working based on user interaction such as:
      i. Checking browser load.
      ii. Validate animations of DIVs where appropriate.

3. HTML
   Verify all tables and images in pages are properly defined.
4.1.2 Procedure

1. Open each browser (Internet Explorer, Mozilla Firefox).

2. For each Web page we will open and execute the following checks
   a. Insure proper look of the interface
   b. Check grammar and spelling.
   c. Insure proper terminology that is commonly understood by all users.
   d. Verify all hyperlinks are valid.

4.1.3 Expected Results

Table 22. Results of the Web Test Cases

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Internet Explorer</th>
<th>Mozilla Firefox</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Stylesheet Verification</td>
<td>Passed</td>
<td>Passed</td>
</tr>
<tr>
<td>2. JavaScript Verification</td>
<td>Passed</td>
<td>Passed</td>
</tr>
<tr>
<td>3. HTML Verification</td>
<td>Passed</td>
<td>Passed</td>
</tr>
</tbody>
</table>
4.2 Testing for Robustness

4.2.1 Purpose

For testing for robustness, we must check to make sure that all pages can handle errors and how the ReMoTe will report the errors to the user.

4.2.2 Procedure

1. Open a browser (Internet Explorer, Mozilla Firefox).
2. Go to each page and purposely put invalid data and/or blank fields and see how the system handles the invalid information.

4.2.3 Results

4.2.3.1 Invalid Login. The most common error users will have on the ReMoTe is supplying invalid login. The ReMoTe will stop and report to the user that the login is invalid (see Figure 59).
Figure 59. Invalid Login

4.2.3.2 Invalid Email. Another common mistake user will have on the ReMoTe is users will supply an invalid email address, which may have typos or is incomplete (i.e. missing the .com). If the error occurs, the ReMoTe will alert them (see Figure 60).
4.2.3.3 Blank Input. Another common mistake user has while using the system is they might not provide information in fields that are required. The ReMoTe checks all of the fields in the form, if a field is left blank, the user cannot continue until the field has been satisfied with the correct information (see Figures 61, 62, and 63).
Figure 61. Message Left Blank on Settings

Figure 62. Note Left Blank on Approval
4.2.3.4 Code as Input. For security purposes, the ReMoTe rejects input that contains any coding, such as Javascript, PHP, etc., which could hurt the system and/or the users using the system. The ReMoTe checks all fields and will not let the user continue using the program if any of the fields in the form contain "<, >, <?,?>, &gt;, &lt;" (see Figures 64 and 65).
Figure 64. Code as Input at Setting Note Page
4.2.3.5 Invalid Hours and Budget Allocation. The ReMoTe also validates forms that require Managers to disperse funding and/or man-hours required for teams. If the amount dispersed to the teams do not equal what the project manager supplied to the user, then the values are rejected and are prompted to try again with correct values (see Figures 66 and 67).
Figure 66. Invalid Hours Allocation
Figure 67. Invalid Budget Allocation

4.2.3.6 Invalid Method Selection. The ReMoTe also checks is drop down box values are correct. For example in Figure 68, the user is to select their team and the methods each teammate will use. If the user does not supply a valid method (life-cycle) to a member, the ReMoTe will reject the input. The user will have to try again with a correct value for the method.
4.2.3.7 Invalid Database Location. In the ReMoTe, the view-only users are required to supply the database locations for each of the projects they want to view. If the user supplies an invalid database location, the ReMoTe will reject the input and alert them to check their input values in the form (see Figure 69).
Figure 69. Invalid Database Location
5.1 Purpose

In order for a proper installation of the ReMoTe tool, the following must be installed: Apache, PHP and mySQL/Oracle/Access. All tutorials in this section are intended for Gentoo operating system.

5.2 Install Apache

Use the following configure command when installing Apache:

```
./configure \
--prefix=/usr/local/apache \
--enable-modules=all \
make && make install
```

After setup has completed start apache in the bin directory:

```
./apachectl restart
```

Next, go to a browser and type host of your new Apache server (e.g. http://www.testpage.com/remote). You must than verify if the Apache default page appears, if it shows an Apache default page, you have completed the first step of getting ReMoTe working.
5.3 Install My-Structured Query Language

In My-Structured Query Language (mySQL) please create a user with permissions to read and write to database. In order for the ReMoTe to work properly, you must set the database location for the ReMoTe to access. Then you must go to your my.cnf file (do a "locate" or "find" depending on OS) Change the server location from 127.0.0.1 to your IP address so it can be remotely connected. Also, you will need to set the max_allowed_packet=1G in the my.cnf file, otherwise the system may not be able to handle certain data correctly.

5.4 Install Oracle

1. In a browser, go to:

   http://otn.oracle.com/software/tech/oci/instantclient
   /htdocs/linuxsoft.html

2. Download the Basic client package with SDK, which are:

   a. oracle-instantclient-basic-10.1.0.3-1.i386.rpm
   b. oracle-instantclient-devel-10.1.0.3-1.i386.rpm

3. Put the RPM files in /usr/portage/distfiles.

4. In the terminal execute:

   ACCEPT_KEYWORDS="-x86" emerge=dev-db/oracle-instantclient-basic-10.1.0.3
5. Finally, execute in the terminal:

   USE="gd odbc oracle oci8" emerge php mod_php

5.5 Install Hypertext Pre-Processor

Before you install Hypertext Pre-Processor (PHP), please make sure that you have the GD graphics library installed, otherwise, you will not be able to view the Gantt charts and statistics. If you receive a result of the corresponding files, you have the correct libraries installed. Otherwise must download and install these RPM files.

5.5.1 Setup Hypertext Pre-Processor with Gentoo Operating System

In the terminal you must execute as root:

   USE="gd apache2" emerge mysql php mod_php gd

5.5.2 Setup Hypertext Pre-Processor with Red Hat 8.0 and Above

   For best results to install PHP, please use version 4.3 or better. Use the following configure command when installing PHP:

   ./configure \n   --prefix=/usr/local/php \n   --with-system-regex \n   --with-config-file-path=/usr/local \n   --with-apxs=/usr/local/apache/bin/apxs \n
101
--disable-debug \ 
--disable-pear \ 
--enable-versioning \ 
--enable-sockets \ 
--enable-track-vars \ 
--enable-ftp \ 
--enable-memory-limit=yes \ 
--enable-bcmath \ 
--with-mysql=/usr/lib/mysql \ 
--enable-exif \ 
--with-gd \ 
--with-jpeg-dir \ 
--with-png-dir \ 
--with-freetype-dir=/usr/lib \ 
--with-zlib-dir=/usr/local \ 
--enable-register-globals \ 
make && make install

If you have the GD library, JPEG library, or PNG library located in different directories, you must modify/point your options to the correct directory. If you are using a library that is used from the install of the OS please use "shared" instead of a directory location.

You must make sure to have the register_globals turned to "off" in the php.ini.
After the install has completed you may restart Apache using the command:

/etc/init.d/apache2 restart

5.6 How to Extract the ReMoTe

You must then extract the file in a location that you can use with Apache to access it from the web browser.

To extract the files, you must execute the following in the terminal:

tar -xvf ReMoTe.<Version Number>.tar

(where the "version number" is the version you have downloaded)

After you have set up the database, please put your database information and administrator username/password in the setup.php, this will help the ReMoTe keep track of what the database for the RMT is located and what the username and password to login to your database.

5.7 Setting Up the ReMoTe

1. Edit "<remote location>/settings/setup.php"
2. Add your mySQL username after '$user=
3. Add your password of the username after '$pwd='
4. Add your host computer name after '$host='
5. Add your databasename you are using after '$db='
6. You can put your company name after '$companyname='
7. Save and exit.

Once you have completed setting up the setup.php file, the ReMoTe manages the database after this.

5.8 Using Concurrent Version System with ReMoTe

One of the new features of the ReMoTe is if you commit a file link to your thread that is in the Concurrent Version System (CVS), the ReMoTe will determine the latest version of the file and display it to whoever wants to view the progress of your thread. First, in order to view the CVS files from the web, you must have ViewCVS or WebCVS installed for the ReMoTe to function correctly.

The version of the CVS used in the Sample ReMoTe has ViewCVS Version 0.9.2 with CVSGraph utility combined so all the user has to do is extract it, change the Repository Link and copy the cgi files to the cgi-bin directory. Due to the naming scheme of the Web Viewable CVS, in order for the ReMoTe to use the CVS correctly you must install CVS so it is in the cgi-bin directory. For example, if you put a link to a CVS file you would use (if using port 80):

http://www.somesite.com/cgi-bin/viewcvs.cgi/filelocation/file.extension
The ReMoTe will accept mostly any version of CVS and
CVS file name (webcvs.cgi, viewcvs.cgi, cvs.cgi, etc.).
The URL can be accepted from any remote server and port
(0-65535) as long as you input the correct URL, for
example:

http://www.somesite.com:7000/cgi-
bin/filelocation/file.extension

5.9 Source Code

The file structure of the ReMoTe is listed in
Appendix A. Also all of the source code is available in an
attached CD.
6.1 Conclusions

ReMoTe can assist software development companies to monitor and manage the software management process. ReMoTe is a Web-based application. It provides easy access to project managers to monitor the project’s progress remotely, anywhere in the world. It is capable of inserting, deleting, updating, selecting and displaying all thread event information on a software project anywhere in the world. [6]

In comparison with other software project management tools, such as Microsoft Project, we compare the differences of each and notice that the ReMoTe has more unique functionalities when it comes to managing a software project. Table 23 displays the differences of the two tools.
<table>
<thead>
<tr>
<th>Microsoft Project 2003</th>
<th>ReMoTe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only can run on Windows platforms.</td>
<td>Runs on any platform (Windows, Linux, etc.).</td>
</tr>
<tr>
<td>Coordinates specific tasks that must be completed within a specific timeframe.</td>
<td>Manages multiple software projects and multiple phases using the RMT lifecycle cycle using a hierarchical design.</td>
</tr>
<tr>
<td>Integrates with MS Office Products (word doc, PowerPoint, excel).</td>
<td>Can use any type of file format to use a an artifact (pdf, word, sxf, etc)</td>
</tr>
<tr>
<td>Assigns one task</td>
<td>Assign Multiple tasks at one time.</td>
</tr>
<tr>
<td>Must convert everything to html to publish to web, such as Gantt Charts, graphs, etc.</td>
<td>Everything is available online.</td>
</tr>
<tr>
<td>Complex to setup time, costs, tasks.</td>
<td>Everything (time, schedule, etc.) is calculated automatically.</td>
</tr>
<tr>
<td>One person handles and defines entire project.</td>
<td>Everyone takes part of the ReMoTe and define what they are responsible for.</td>
</tr>
<tr>
<td>Does not utilize 3rd party debugging software.</td>
<td>Uses Bugzilla. It can also include other debugging tools (with some coding).</td>
</tr>
<tr>
<td>Does not use any 3rd party File (Software) Repository software.</td>
<td>Uses ViewCVS to view current files being used in the project. It can also include other Repository software (with some coding).</td>
</tr>
<tr>
<td>Project can only Share project planning with others in the organization by saving it to a central workspace.</td>
<td>Everyone on the project can share his or her software artifacts with any other member in the project (and other projects) on the web (any browser, any OS).</td>
</tr>
<tr>
<td>It handles one project at a time, cannot initialize a new iteration of a project.</td>
<td>It handles multiple iterations of a project.</td>
</tr>
<tr>
<td>It cannot determine who in the project is holding down production.</td>
<td>You can view in real-time who is not completing their task.</td>
</tr>
<tr>
<td>Cannot communicate with employee about a file (Resource).</td>
<td>The ReMoTe has its own message board and &quot;Note taker&quot; to allow groups from different parts of a project to alert and/or inform a person about an artifact supplied to their thread.</td>
</tr>
<tr>
<td>Has no individuality of people involved in the project. You cannot find a person’s contact information if there is any notification needed.</td>
<td>Each person in the project can define his or her phone number, location (address), in case anyone in the project needs to contact him or her.</td>
</tr>
<tr>
<td>It cannot give an actual progress</td>
<td>The ReMoTe can give each person’s progress.</td>
</tr>
</tbody>
</table>
Using CVS linkage enables the support for teamwork and management of huge and complex software. With ReMoTe, the changes at any project level can be seen immediately. The individuals as well as teams progress can be calculated in term of percentage. At the time of writing, ReMoTe supports Oracle, MySQL, and Microsoft Access, but since the multi-database has been built into ReMoTe, it is relatively a simple task to have other database system added such as PostgreSQL, Sybase, OpenBase, FrontBase, DB2, etc.

6.2 Future Directions

Functions that can be added in the future are a function that can manage multiple projects with procedure relation. ReMoTe can calculate the longest time (delivery time) to complete the entire project. There is precedence relation in multiple projects, where certain project cannot start until all the projects that have precedence
over it are all finished. The management of multiple projects is calculated using critical path analysis algorithm so that the time to complete such sizable multiple projects can be determined.

Another function that can be added to ReMoTe is a client version of the program, which allows users to add groups of software artifacts at a time instead of individually uploading files one by one. The client should be able to run on both Linux and Windows platforms.
APPENDIX A

FILE STRUCTURE
ReMoTe //root directory
|-- DOCS //documentation files
  |-- CHANGES
  |-- TODO
  `-- masters
     |-- PreliminaryProposalFormat.doc
     |-- ReMoTe_SRS.doc
     `-- Summer.doc
|-- GanttChart.php
|-- README.txt //overall description of installation
|-- ShowImage.php
|-- WEB-INF //source code
  |-- html //interface code
     |-- adminPages //administrator interface
        |-- acceptUser.inc
        |-- createMessage.inc
        |-- createProject.inc
        |-- deleteUser.inc
        |-- editMessages.inc
        |-- editRegistration.inc
        |-- editUser.inc
        |-- intro.inc
        `-- layout //administrator template files
            |-- admin.tpl
            `-- logout.tpl
     |-- login.inc
     `-- register.inc
|-- clientPages //user interface
     |-- approveThreads.inc
     |-- costAnalysis.inc
     |-- createMessage.inc
     |-- createThread.inc
     |-- defineThread.inc
     |-- editMethod.inc
     |-- editRegistration.inc
     |-- editViewProjects.inc
     |-- ganttChart.inc
     |-- intro.inc
     `-- layout //user template files
        |-- client.tpl
        `-- logout.tpl
            |-- redirect.tpl
     `-- login.inc
     `-- noteList.inc
     `-- password.inc
     `-- redirectAdmin.inc
     `-- register.inc
|-- selectTeam.inc
|-- setCost.inc
|-- setDates.inc
|-- setHours.inc
|-- setNumberPhases.inc
|-- setPicture.inc
|-- setProjectWeights.inc
|-- setTeamNames.inc
|-- setViewProjects.inc
|-- settings.inc
|-- statistics.inc
|-- viewMessage.inc
|-- viewMessages.inc
--+ viewProject.inc

|-- threadPages //thread output interface
|-- compress.inc
|-- createNotes.inc
|-- cvsgrab.inc
|-- cvsgrab.php
|-- cvsoutput.php
|-- ganttChart.inc
|-- rugzilla.php
|-- showFile.inc
|-- showFile.php
|-- showPhase.php
|-- showTeamList.inc
|-- statistics.inc
|-- threadLayout
| |-- bugzilla.tpl
|`-- thread.tpl

|-- src //php source code
|-- admin //administrator action files
| |-- AcceptUserAction.php
| |-- CreateProjectAction.php
| |-- DeleteMessageAction.php
| |-- DeleteProjectAction.php
| |-- DeleteUserAction.php
| |-- EditMessageAction.php
| |-- EditMessagesAction.php
| |-- EditUserAction.php
| |-- EditUserPageAction.php
| |-- LoginAction.php
| |-- RegistrationAction.php
| |-- SetProjectManagerAction.php
|`-- SetViewOnlyAction.php

|-- client //user action files
-- Chat.php
-- ChatDAO.php
-- ChatUpload.php
-- ChatUploadBlob.php
-- ChatUploadDAO.php
-- ComputeStats.php
-- ConnectDAO.php
-- Cost.php
-- CostDAO.php
-- Database.php
-- DefineThread.php
-- DefineThreadDAO.php
-- FileNotes.php
-- FileNotesDAO.php
-- Hours.php
-- HoursDAO.php
-- MessageBoard.php
-- MessageBoardDAO.php
-- Note.php
-- NoteDAO.php
-- ODBCSocketServer.php
-- Project.php
-- ProjectDAO.php
-- SelectTeam.php
-- SelectTeamDAO.php
-- SetMethod.php
-- SetMethodDAO.php
-- SetNumberPhases.php
-- SetNumberPhasesDAO.php
-- SetProjectWeights.php
-- SetProjectWeightsDAO.php
-- SetTeamNames.php
-- SetTeamNamesDAO.php
-- SetViewProjects.php
-- SetViewProjectsDAO.php
-- Settings.php
-- SettingsDAO.php
-- Stats.php
-- StatsDAO.php
-- ThreadDAO.php
-- URL.php
-- User.php
-- UserDAO.php
-- urlDAO.php
-- thread //thread output action files
  -- GanttChartAction.php
  -- ShowFileAction.php
-- ShowItem.php
-- ShowNotesAction.php
-- ShowTeamListAction.php
-- StatisticsAction.php

-- util
-- DebugLogger.php

-- static // non-changing files i.e. images, scripts
-- css // stylesheets
-- admin.css
-- clientDefault.css

-- help // user help folder
-- About.php
-- README.html
-- frameMain.html
-- helpFiles
-- README.html
-- ApproveThreadsSmall.jpg
-- Chatroom.html
-- CreateCvs.html
-- Cvstutorial.html
-- Deadlines.html
-- images
-- GanttChart.jpg
-- accept.jpg
-- addmessage.jpg
-- addthread.jpg
-- addthread1.jpg
-- adminlogin.jpg
-- adminside.jpg
-- ApproveThreadsSmall.jpg
-- change.jpg
-- chat.jpg
-- colorsheme.jpg
-- contact.jpg
-- CreateMessageSmall.jpg
-- Database.jpg
-- Deadlines.jpg
-- DefineThreadSmall.jpg
-- EditThread.jpg
-- Engineers.jpg
-- ForgotSmall.jpg
-- Intro.jpg
-- IntroSmall.jpg
-- Intro_meeting.jpg
-- LoginSmall.jpg
-- LogOut.jpg
-- MainManager.jpg
REFERENCES


