

UberCMS

A Unique Solution for Software Engineers

Version 0.75

UberSoft

March 2006



Executive Summary

This document describes the architecture and design principles that will be used to create and implement the UberCMS system. It goes into detail on the context and reasons for creating the UberCMS as well as the features of the different interfaces. It covers the different components used in the structure and the dynamic behaviour of the system. The document also explains the possible change scenarios and how the UberCMS development team will deal with those issues. We discuss the possibility of reusing code for different sections of UberCMS and the reasons for selecting these sections. The document explains UberCMS' approaches to Standard Functional Areas and gives a requirements traceability matrix to help define the subsystems. There is also an explanation of the support for staged delivery that will be used.

The context of UberCMS is to help the creation and organization of Software Engineering documents and to facilitate communication between group members. There are three different interfaces which will be used: one for a user, one for a project leader, and one for the administrator.

The architecture decided upon for the UberCMS system is PHP and MySQL which are available on Beddie, a server run by the provided by the Computer Science Club. The dynamic behaviour section illustrates the flow of the UberCMS architecture using sequence diagrams.

UberSoft has decided to reuse code for the forums as well as the rtf editor. Code was also found to create the rtf editor and the forums to help in the creation of the UberCMS system. The subsystems being used for this project are an Apache, file storage, a Database and UberCMS.

TABLE OF CONTENTS

1. INTRODUCTION	- 5 -
1.1 Authors	- 5 -
1.2 Creation and Modification History`	- 5 -
1.3 Audience	- 6 -
1.4 Purpose of this Document.....	- 6 -
1.5 Related Documents	- 6 -
2. SYSTEM PURPOSE.....	- 7 -
2.1 UberCMS Purpose and Context	- 7 -
2.2 System Interface Section	- 7 -
2.3 Non-Functional Requirements Section	- 11 -
3. STRUCTURE.....	- 13 -
3.1 Overview.....	- 13 -
3.2 Components	- 15 -
4. DYNAMIC BEHAVIOR	- 17 -
4.1 Scenarios	- 17 -
5. CHANGE SCENARIO AND CHANGE STRATEGY	- 26 -
5.1 Risks.....	- 26 -
5.2 Addressing changes	- 27 -
6. REUSE ANALYSIS	- 28 -
7. APPROACHES TO STANDARD FUNCTIONAL AREAS	- 29 -
7.1 Database Organization	- 29 -
7.2 Data storage	- 30 -
7.3 Key Algorithms.....	- 30 -
7.4 Memory Management.....	- 30 -

7.5 String Storage	- 31 -
7.6 Concurrency and threads	- 31 -
7.7 Security.....	- 32 -
7.8 Localization	- 32 -
7.9 Networking.....	- 32 -
7.10 Portability.....	- 33 -
7.11 Programming Language	- 33 -
7.12 Error Handling	- 33 -
7.13 Documentation	- 33 -
8. REQUIREMENTS TRACEABILITY & TESTING.....	- 34 -
9. SUPPORT FOR STAGED DELIVERY	- 36 -
9.1 Procedure for Release	- 36 -
9.2 Stage 1 – Critical Components.....	- 36 -
9.3 Stage 2 – File control and manipulation	- 37 -
9.5 Stage 4 – Future Enhancements	- 38 -
10. CONCLUSION.....	- 39 -
APPENDIX A.....	- 40 -

1. Introduction

The architecture being used in this document is called UberCMS.

1.1 Authors

The people involved in the creation of this document and their responsibilities are:

Author	Responsibilities
Devon Austen	Dynamic Behaviour, Conclusion, Editing
Dan Clark	Introduction, System Purpose, Executive Summary
Amy Gill	Approaches to standard functional areas, Final editing.
Jason Mansfield	Reuse Analysis, Requirements traceability, Conclusion
Andrew McNally	System Structure, Editing.
Jessica Reyes	Change Scenario and Change Strategy, Support for staged delivery, Editing

1.2 Creation and Modification History`

Version	Date	Description	Authors
0.55	March 14 th	Created high level outline of what needed to be done.	UberSoft
0.6	March 15 th	Created sequence diagrams, System purpose, Code Reuse, Notation, Approaches to standard functional areas.	UberSoft
0.65	March 17 th	Created Introduction, Support for staged delivery, Requirements Traceability	UberSoft
0.7	March 19 th	Created Change Scenario and Change Strategy, Conclusion	UberSoft
0.75	March 21 st	Joined all of the separate parts together and reviewed the document fixing grammar and spelling.	UberSoft

1.3 Audience

The stakeholders involved in this document are numerous, they are as follows:

Stakeholders	Justification
Developers of the UberCMS system	They will directly use this document to help further understanding and creation of the system.
Managers of UberSoft	They will read this to make sure UberCMS is following what the company had in mind when they started the project.
Software Engineers	They could be asked to look over the document to make sure that it follows what they need in a CMS for Software Engineering documents.
Marketers	They need to know and understand what the system will do so they can properly and effectively market it to the public
People funding the project	They want to know what their money is doing and to make sure that their investment will pay off.

1.4 Purpose of this Document

The UberCMS is a system designed to help facilitate the creation and management of software engineering documents. The main focus is to cater to the needs of software engineers during the design phase of program creation. This architecture document will help to explain the functionality of UberCMS and how it will be implemented. It will explain the constraints on the system as well as the various problems that could crop up during implementation and how to solve them. After reading this document, the reader should have a good understanding of what UberSoft's goal is for the UberCMS and how they plan to achieve this goal.

1.5 Related Documents

For further information regarding UberCMS, please read the UberCMS System Requirements Specification (SRS) document.

2. System Purpose

The System Purpose section outlines the reasons for creating the UberCMS system. It gives a high-level description of what UberSoft has in mind for the UberCMS system. It also goes into detail about what each interface within the system will do, and what constraints will be placed on the system.

2.1 UberCMS Purpose and Context

In the design process of any large computer application, software engineers are required to create many different types of documents. Examples of these documents are: System Requirements Specification, architecture documents, and sequence diagrams. These documents are very important in the creation of applications as they allow the developers to minimize the time they spend thrashing around trying to program. The chances of completing construction of an application dramatically increase with the use of well organized design documents.

UberCMS is an application that can manage and organize the various types of software engineering documents. E.g. users can upload, store and modify the various documents on the UberCMS server. UberCMS will store the documents according to the different projects and allow users only to access their designated projects.

2.2 System Interface Section¹

UberCMS User Interface

Interface	UberCMS User Interface
Use Cases	UC1 Edit a file: This service allows users to edit rtf files in the built in rtf editor. All of this can be done within UberCMS without having to open a new program.
	UC2 Upload a file: This service allows users to upload files, which may not be supported for editing in UberCMS, from their computer to UberCMS. The file will be time stamped, stored in the designated folder and any previous versions will be archived.

¹ For further information about use cases, please reference the UberCMS SRS document

	<p>UC3 Download a file: This service allows users to download files from the UberCMS to their computer to be edited.</p>
	<p>UC4 Save a file: This service allows users working on files within the built-in rtf editor to save the files when they are finished. As a result, the old files will be archived and replaced with the new, time-stamped file.</p>
	<p>UC5 Select an archived file: This service allows users to view, and edit old versions of files either by downloading to their pc, or by using UberCMS' rtf editor</p>
	<p>UC6 Open a file in the Editor: This service allows users to open rtf files in the built-in rtf editor.</p>
	<p>UC7 Create a new file: This service allows users to create new rtf files in the editor.</p>
	<p>UC8 Select a requirements template: This service allows users to select a requirements template format to follow in the rtf editor.</p>
	<p>UC9 Select a use case template: This service allows users to select a use case template format to follow in the rtf editor</p>
	<p>UC25 Select a project: This service allows users to select which project they wish to enter from a list of available projects</p>
	<p>UC26 Look at archived announcements: This service allows users to look at old announcements that may not be viewable from the announcements section.</p>
	<p>UC27 Select menu options: This service allows users to navigate through UberCMS using a menu system.</p>

	<p>UC28 Select an event from the calendar: This service allows users to select an event from the calendar section to be able to view further information about the event.</p>
	<p>UC29 Create a new thread: This service allows users to create a message on the forum.</p>
	<p>UC30 Edit post: This service allows users to edit a post that they created.</p>
	<p>UC31 Email all users: This service allows users to email all the users in a specific project.</p>
	<p>UC32 Email selected users: This service allows users to email only certain users by clicking the checkbox beside their name.</p>
	<p>UC33 Log in: This service allows users to access the UberCMS with a login name and password.</p>
	<p>UC34 Reply to a previously written message: This service allows users to post a message in response to another message.</p>
	<p>UC35 Select a thread: This service allows users to view the messages contained within a thread.</p>
	<p>UC36 Log out: This service allows users to leave the UberCMS system.</p>

UberCMS Project Leader Interface²

Interface	UberCMS project leader interface
Use Cases	<p data-bbox="532 415 1383 527">UC10 Move a file from one directory to another: This service allows the project leader to organize the files contained within the system.</p> <p data-bbox="532 600 1370 669">UC11 Create a directory: This service allows the project leader to create new directories to store files.</p> <p data-bbox="532 743 1360 812">UC12 Delete a directory: This service allows the project leader to remove directories that are no longer needed.</p> <p data-bbox="532 886 1300 955">UC13 Rename a directory: This service allows the project leader to rename directories that may have been misnamed.</p> <p data-bbox="532 1029 1341 1140">UC14 Add an announcement: This service allows the project leader to add announcements to the announcement section that can be viewed by everyone in the project.</p> <p data-bbox="532 1213 1377 1283">UC15 Add an event or due date: This service allows the project leader to add an event or due date to the calendar.</p> <p data-bbox="532 1356 1312 1446">UC18 Delete posts: This service allows the project leader to remove posts from the forum that are no longer needed.</p> <p data-bbox="532 1520 1321 1589">UC23 Unlock a file: This service allows the project leader to unlock a file that no longer needs to be locked.</p> <p data-bbox="532 1663 1357 1732">UC24 Lock a file: This service allows the project leader to lock files that they don't want people to be able to change.</p>

² Please note: The project leader can also do everything that a user can.

UberCMS Administrator Interface³

Interface	UberCMS administrator interface
Use Cases	UC16 Add a project: This service allows the Administrator to create new projects that can be viewed and worked on by the other users.
	UC17 Add a user: This service allows the Administrator to add a user to a desired project with all of the appropriate access rights.
	UC19 Delete a project: This service allows the Administrator to delete projects that are no longer needed.
	UC20 Delete a user: This service allows the Administrator to delete users from projects or from the UberCMS system itself.
	UC21 Assign a project leader: This service allows the Administrator to create a project leader, a person with similar access rights as the admin, except only in their designated project.
	UC22 Modifying a user's access rights: This service allows the Administrator to restrict access to section of UberCMS to different users.

2.3 Non-Functional Requirements Section

UberCMS: Qualities

Security:

- Only one person will be able to modify a file at any one time.
- There will be three different levels of users to restrict access to different parts of the system. The administrator will have full access to entire system and be able to change other user's permission levels.

³ Please note: The Administrator can do everything that the project leader can do as well as the user.

- When a file is modified, the original copy will be archived on the web server. This will protect users from accidental deletion or malicious changes to files.

Performance:

- The system will store files on a web server.
- The system will store user information in a Database contained within Beddie.

UberCMS: Constraints

- The system must run in the Mozilla Firefox and Internet Explorer browsers.
- The system will register changes to files as soon as the files are saved.

UberCMS: Principles

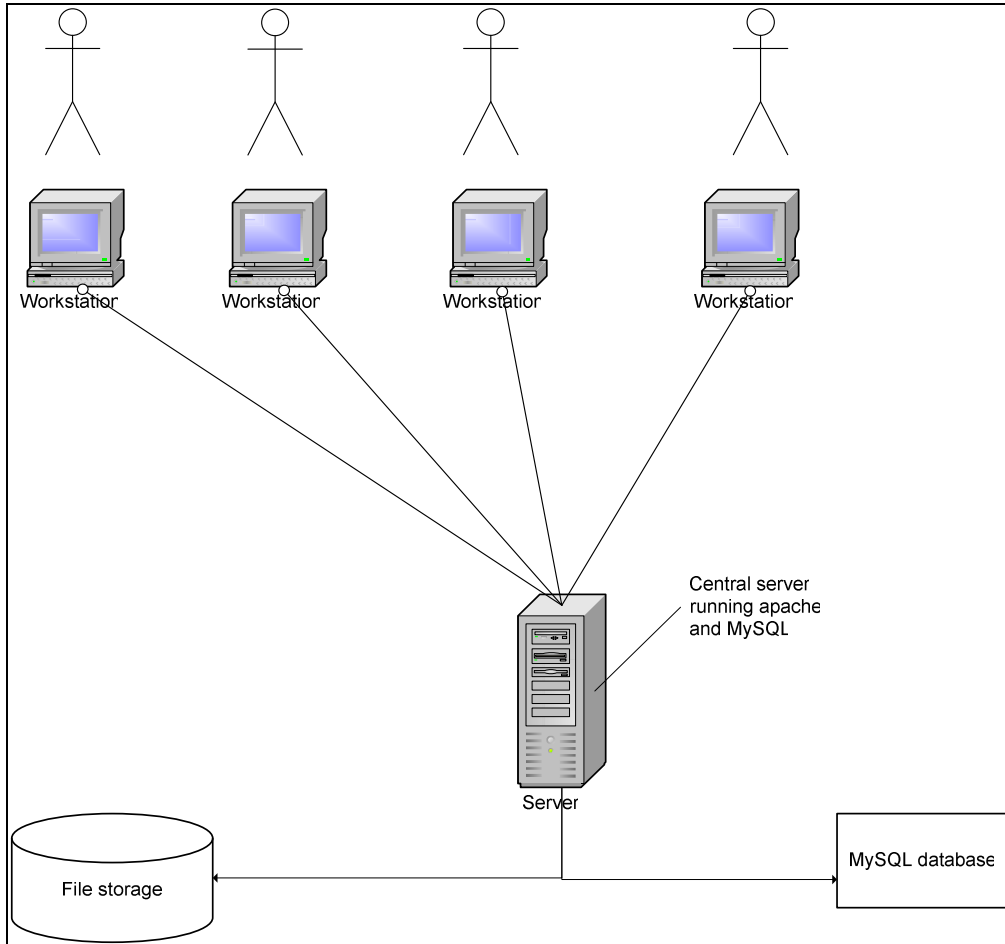
- The system will reuse code for the rtf editor as well as the forums.
- Each page will contain the same menu system for easy browsing of the UberCMS system.
- Only rtf files will be editable from within the UberCMS system. All others will need to be downloaded and modified in their native program.
- The system will provide different templates for design documents within the rtf editor.

3. Structure

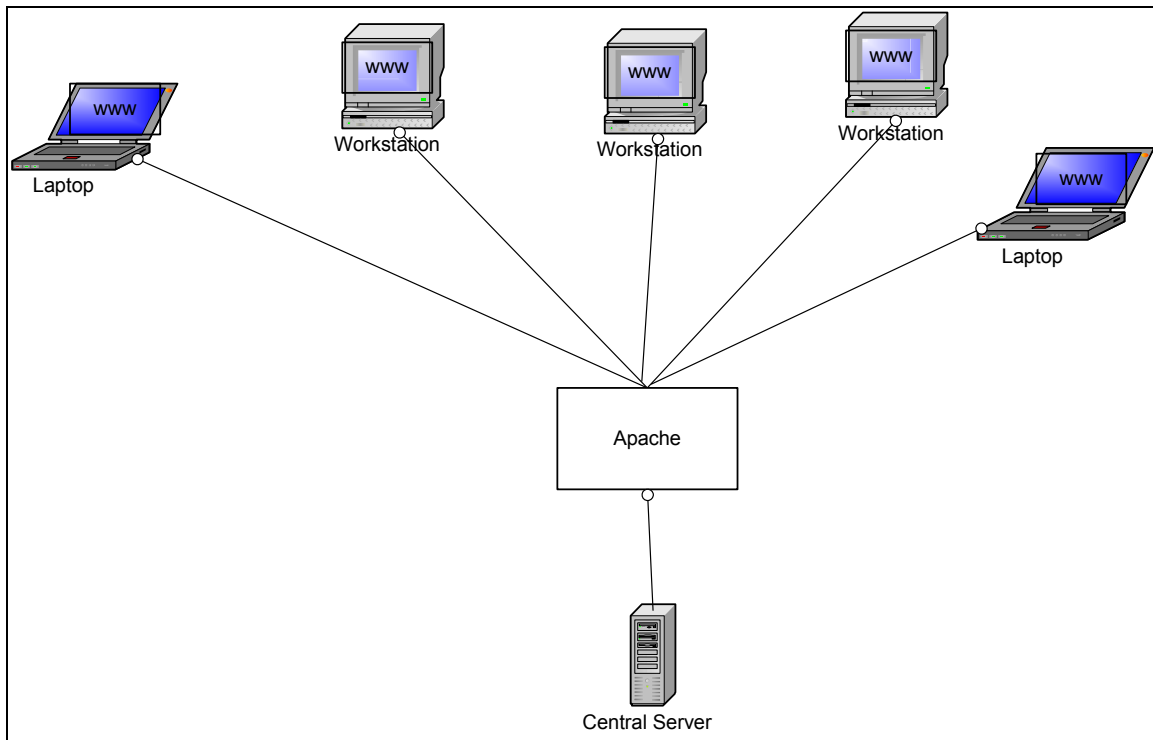
The following section describes the architectural design of UberCMS. There are three diagrams, the high-level architecture, the front-end structure and the back-end structure; these diagrams are followed by a description of the components of UberCMS.

3.1 Overview

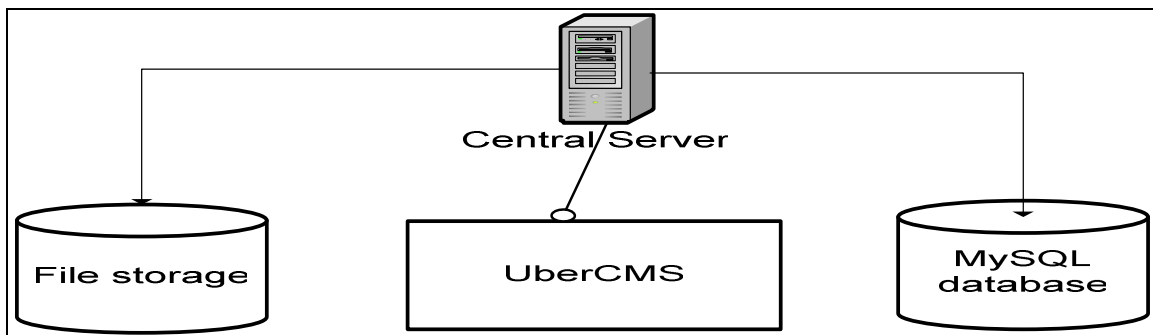
Architectural Diagram



Front-end Architecture



Back-end Architecture



Chosen Architecture

After considering several different architectural designs for UberCMS, we at UberSoft ultimately decided to use PHP to implement the web interface and use a MySQL database and a file server for the storage of all data.

The decision to use a MySQL database and file server for storage was agreed upon because our development team had easy access to a fully functioning file server that already had a MySQL server running on it. This allowed us to forgo the setup and configuration time that may have been needed had we chosen an alternative architecture. If money had been a concern this decision would allow UberSoft to save on the costs of any new hardware or software that would be required in possible alternate architectures.

The decision to use PHP to implement the web interface was made for several reasons. First, and most important was, some of members of the development team already knew how to use PHP so there would be no need to learn the language in order to use it in our implementation. Second, we used PHP because it is a fairly powerful language and allows for easy integration with the other components of the architecture that we decided upon.

Alternative Architectures

During our discussions about the desired architecture, we came up with two alternative designs to the one we decided to implement. The first one was similar in that it used PHP for the web interface, but was different in the fact that it would have used XML files instead of the MySQL database. We decided against this design because none of our development team had any experience working with XML files and felt that sticking with something they were more comfortable with would produce better results. The other architecture design that we considered was to use Ruby on Rails for the web interface and then either XML files or a SQL database and a file server for the data storage. We decided against this design path as we felt the learning curve of Ruby on Rails was too great for the time we had to design and implement UberCMS. After considering all options we decided on the one that the development team was most comfortable with as they are the people that will eventually have to implement the system.

3.2 Components

Component	MySQL database.
Responsibilities	Used to store all of the information pertaining to users and projects with the exception of the project files, which will be stored separate location. This information stored in the database will include, all active user's info, active projects and announcements, calendar and contacts for all projects.
Collaborators	None
Notes	There will be one instance of the MySQL database running on the central server. As we are using an existing version of MySQL, there is concurrency controls natively built into it. Additionally, we will make daily backups of all databases to help insure minimal data loss in the case of system failure.
Issues	None

Component	Apache web server.
Responsibilities	The interface between UberCMS and the database and file storage.
Collaborators	File storage, MySQL database
Notes	There will be one instance of apache running on the central server. There will be no concurrency control as all apache does is control the web interactions.
Issues	None

Component	File Storage
Responsibilities	Location where all project files are organized and stored.
Collaborators	None
Notes	There is only one instance of this component, and it is restricted by the file storage capacity of the central server. All of the files contained in a project must remain available as long as the project they belong to exists. The files will be protected with a CVS like file locking system to prevent editing of a file by more than one person simultaneously and all files will be backed up daily to prevent data loss in the case of a disaster.
Issues	None

Component	UberCMS
Responsibilities	The main interaction between the user and the central server.
Collaborators	Apache Server, Database, File Storage
Notes	As UberCMS is a web based program, there is one instance of the program files located on the central server and multiple users can access UberCMS using a web browser.
Issues	None

Table 1: Component specifications

4. Dynamic Behavior

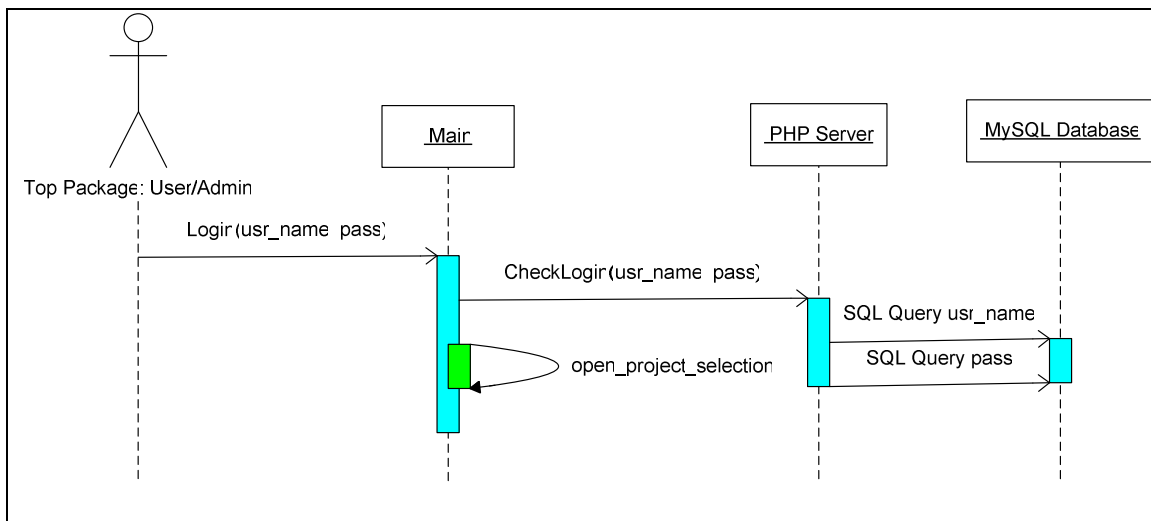
This section will explore the essential use cases or scenarios of the system and explain their behaviour within the context of our system architecture. It will do so first by describing the scenario and then it will show the behaviour within the context of the architecture through a sequence diagram.

4.1 Scenarios

Scenario Specification: Worker Login

Use Case	Log in	#33
Description	Allows workers to login to the system.	
Actors	Users(primary) Project leaders Administrator	
Steps	<ol style="list-style-type: none"> 1. Worker can enter a username and password. System only allows valid users to proceed. 2. System opens project selection page. 	

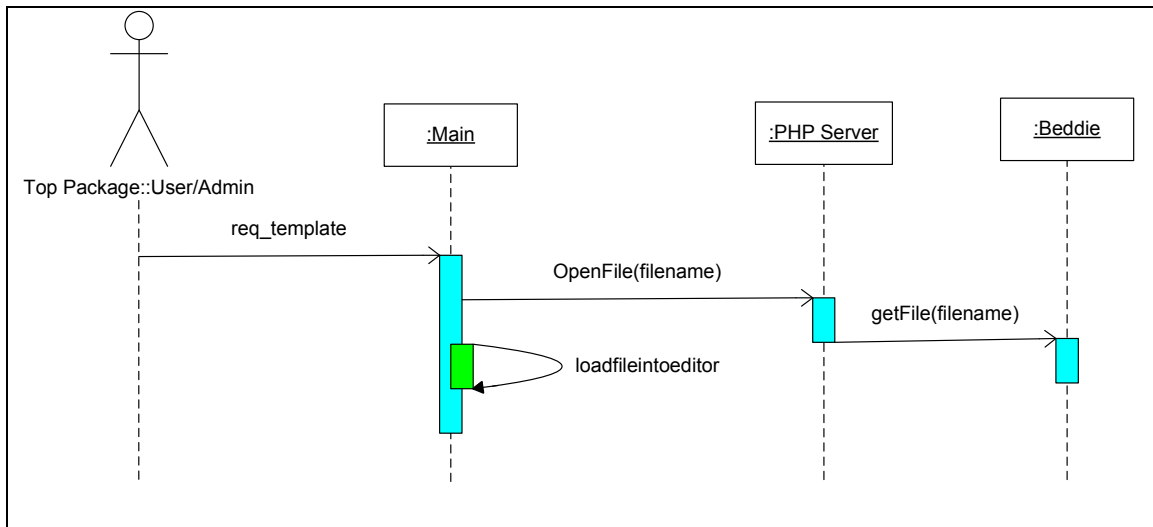
Component Interaction Model: Worker Login



Scenario Specification: Worker Selects a Requirements template

Use Case	Select a Requirements template	#8
Description	The template is inserted into the file that the worker is working on.	
Actors	Users(primary) Project leaders Administrator	
Steps	1. Worker presses the requirements template button. 2. System loads the template in RTF editor.	

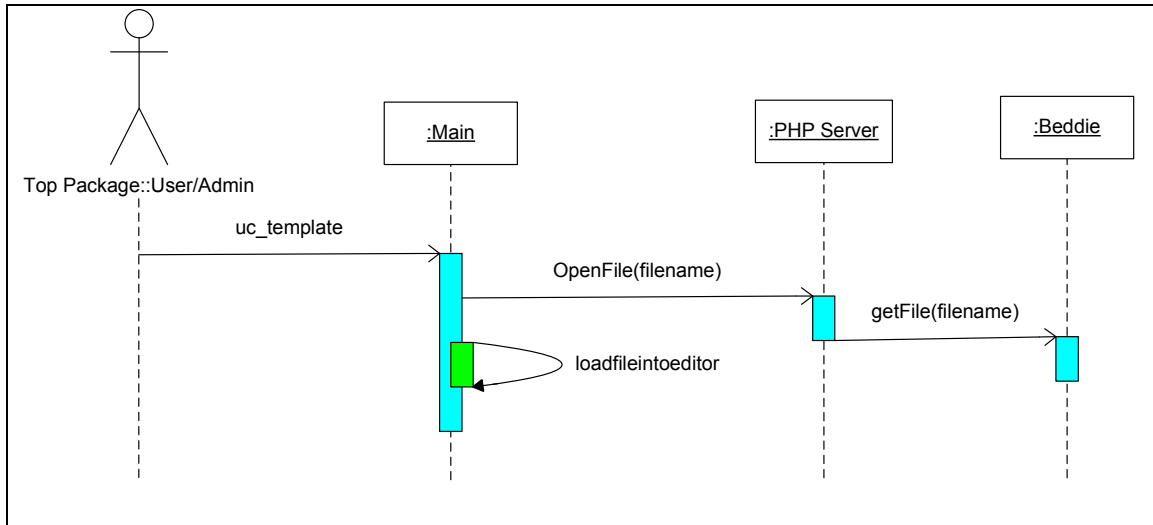
Component Interaction Model: Worker Selects a Requirements template



Scenario Specification: Worker Selects a Use Case Template

Use Case	Select a use case template	#9
Description	The template is inserted into the file that the worker is working on.	
Actors	Users(primary) Project leaders Administrator	
Steps	1. Worker presses the use case template button. 2. System loads the template in RTF editor.	

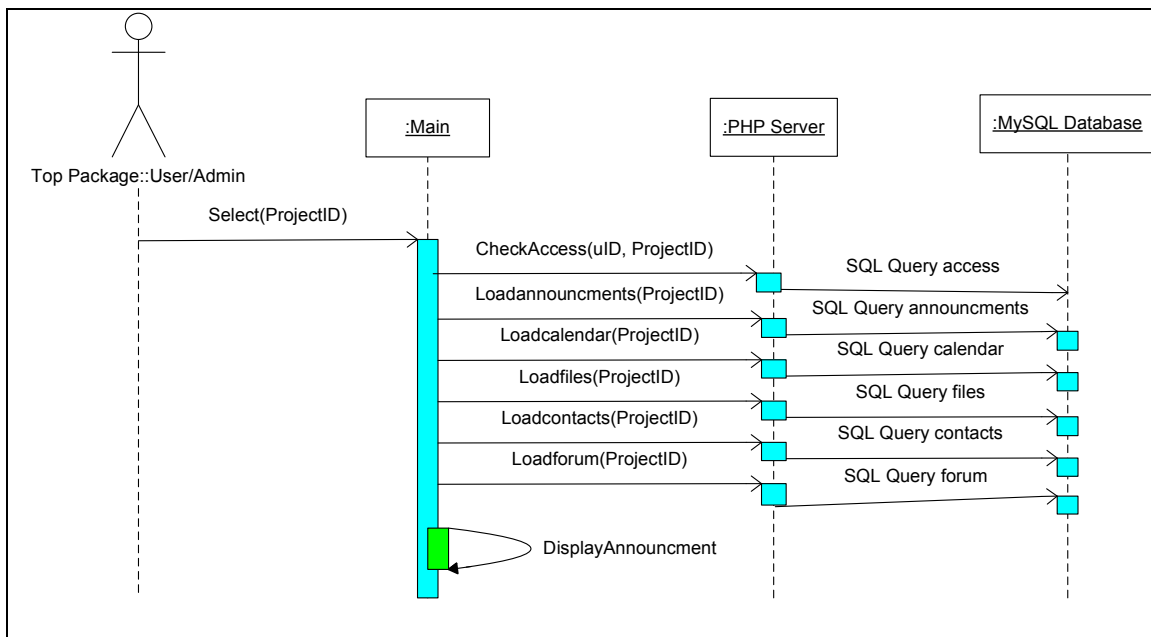
Component Interaction Model: Worker Selects a Use Case Template



Scenario Specification: Worker Selects a Project

Use Case	Select a Project	#25
Description	Logs into a specific project and display announcements page.	
Actors	Users(primary) Project leaders Administrator	
Steps	<ol style="list-style-type: none"> 1. Worker clicks on a specific project image and system loads project from database. 2. System displays the announcement page. 	

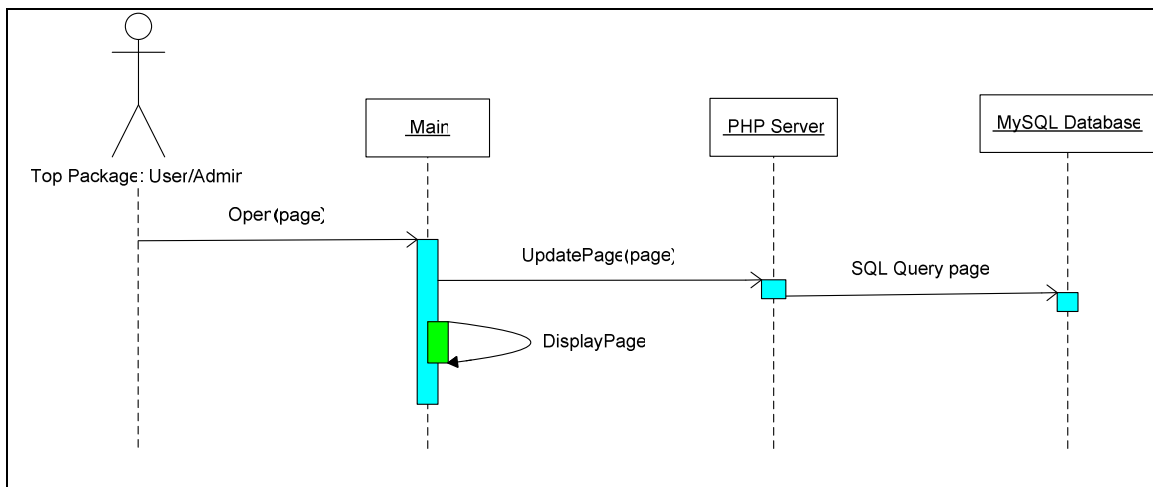
Component Interaction Model: Worker Selects a Project



Scenario Specification: Worker Selects a Menu Item

Use Case	Select a Menu Option	#27
Description	Upon Select a menu item the system loads corresponding page.	
Actors	Users(primary) Project leaders Administrator	
Steps	<ol style="list-style-type: none"> worker clicks on a specific menu item and system updates from database selected page. System displays the selected page. 	

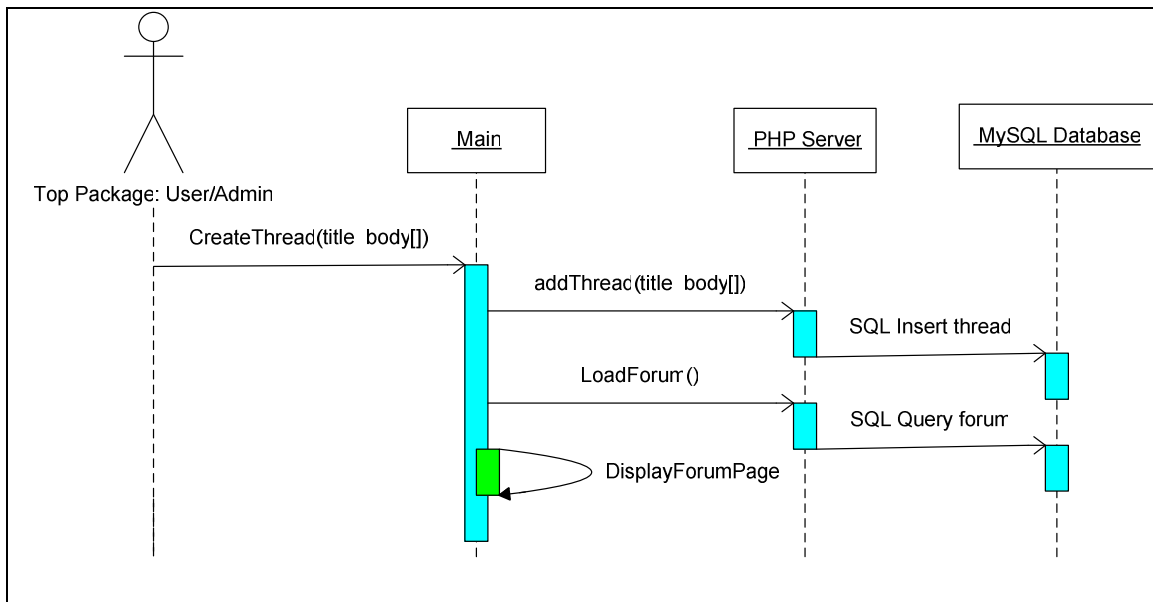
Component Interaction Model: Worker Selects a Menu Item



Scenario Specification: Worker Creates Forum Thread

Use Case	Create a new thread	#29
Description	Add a new thread to the forum.	
Actors	Users(primary) Project leaders Administrator	
Steps	<ol style="list-style-type: none"> 1. Worker clicks on submit in the create a new thread page system adds thread to database. 2. System updates forums. 3. System displays forum page. 	

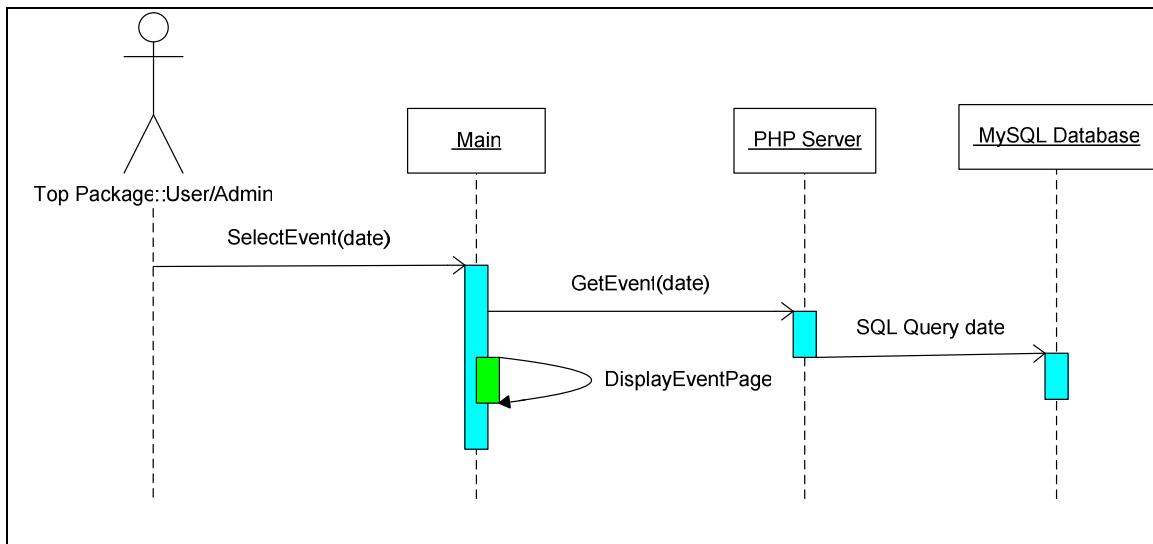
Component Interaction Model: Worker Creates Forum Thread



Scenario Specification: Worker Selects Event from Calendar

Use Case	Select a Event from the Calendar	#28
Description	View detailed information about events or due dates for a selected date.	
Actors	Users(primary) Project leaders Administrator	
Steps	1. Worker clicks on specific date system loads event from database. 2. System displays event page.	

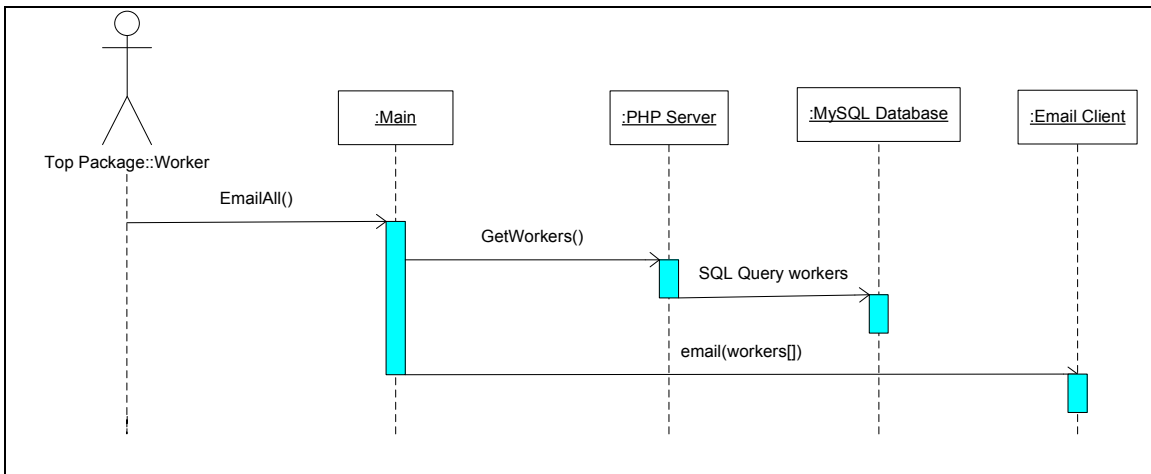
Component Interaction Model: Worker Selects Event from Calendar



Scenario Specification: Worker Emails All Project Workers

Use Case	Email all users	#31
Description	Opens workers native email client with all associated workers in the current project as recipients.	
Actors	Users(primary) Project leaders Administrator	
Steps	<ol style="list-style-type: none"> 1. Worker clicks on email all button system retrieves list of all associated workers. 2. System opens naive email client and tells client who to send the message to. 	

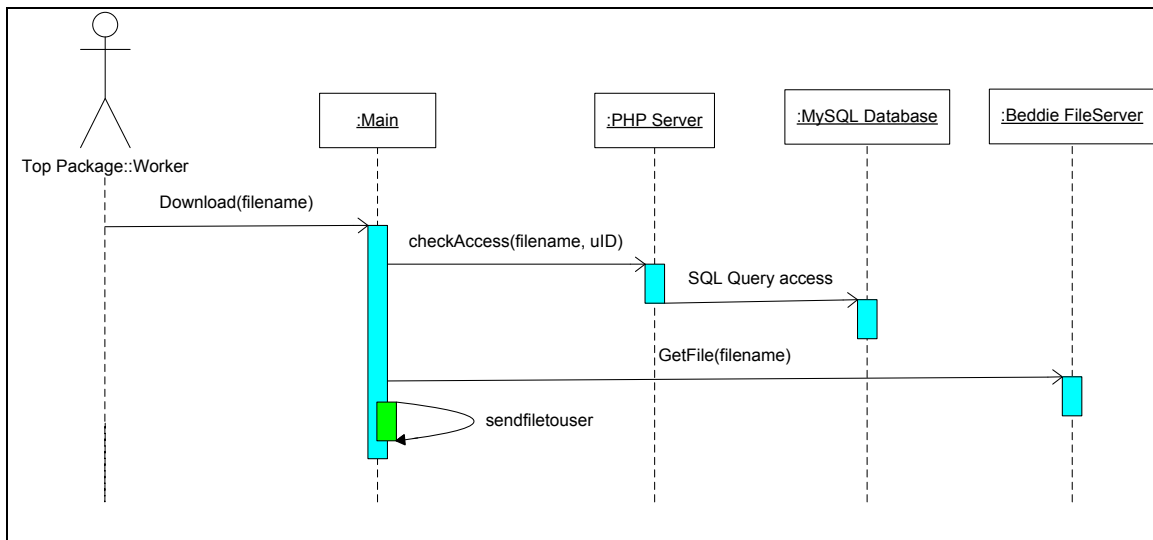
Component Interaction Model: Worker Emails All Project Workers



Scenario Specification: Worker Downloads a File

Use Case	Select a file to download	#03
Description	A selected file is accessed for local download by a worker.	
Actors	Users(primary) Project leaders Administrator	
Steps	<ol style="list-style-type: none"> 1. Worker clicks on download button for desired file system checks to see if worker has access to the file. 2. System gets file from database. 3. System gives worker file for download. 	

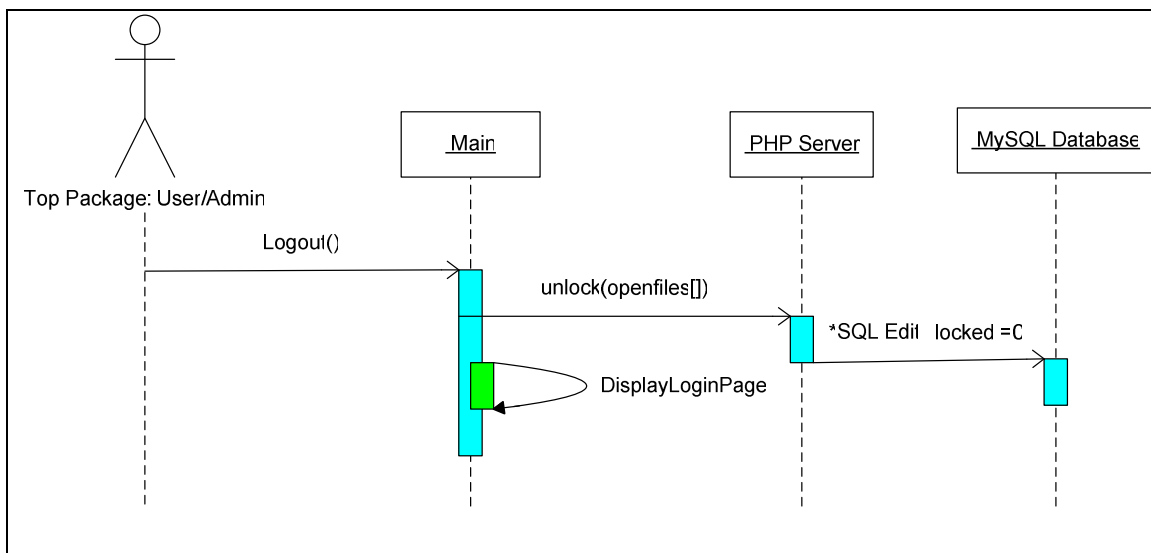
Component Interaction Model: Worker Downloads a File



Scenario Specification: Worker Logout

Use Case	Logging out	#36
Description	Exits worker from system and system loads login page.	
Actors	Users(primary) Project leaders Administrator	
Steps	<ol style="list-style-type: none"> 1. Worker clicks on the logout button and system unlocks currently open files. 2. Exits you from system and reloads login page. 	

Component Interaction Model: Worker Logout



5. Change Scenario and Change Strategy

5.1 Risks

These are the risks that have been identified to this stage of development:

- i. Under allocation of physical resources
 - larger physical storage needed on server than it provides
 - greater number of connections needed at one time
 - central server goes offline

- ii. Difficulties with programming language
 - not adequate for all needs
 - overly complicated for our development team
- iii. Change of personnel
 - reassignment of developers to different tasks due to prior developer responsibilities
 - lost time to get new project members up to speed on unfinished tasks
- iv. File corruption
 - lost files
 - incompatible formats
 - mix-up of file versions
- v. Time constraints
 - overall time constraints due to other projects overlapping with this one
 - final deadlines for release
 - inadequate time allocated for testing
- vi. Requirements re-specification
 - inadequate specification for sub-modules
 - unforeseen problems

5.2 Addressing changes

Based on the above risk assessment, these are some of the strategies we will employ preemptively or in the case of the scenarios occurring:

- i. During the development phase, possible options would be to host our testing database on a developer's personal server, or to look for a new low-cost hosting elsewhere. Hosting expansion will be addressed before expected capacity. Back-up of all files will be compiled nightly and stored on a second location, namely a developer's personal computer.
- ii. Developers with the most PHP and MySQL experience will be assigned lead positions, and delegate appropriate tasks to less experienced members as appropriate. If necessary, javascript or java may be introduced if suited for sub-task development. Less experienced group members will be responsible for learning the skills as needed.
- iii. To prevent backtracking when developers change assignments, developers will be responsible for fully documenting code as construction proceeds. This includes all commenting, pseudo code, and function parameter specification.
- iv. At least two copies of all code will be stored in different locations, one on the CS-Club server, and another on the project leader's system. Written documentation will be stored in .doc files so all developers will be able to access the information. The last two working versions of the software will also be stored in the case of accidental corruption. The project lead will be responsible for versioning the backup files.

- v. In the case of case of time constraints impeding our releases, only critical systems and basic usage will be focused on so that the product achieves high standards on what it does accomplish. Current milestones and timelines will be readjusted accordingly. Testing will be given a higher priority, and allocated more time over other phases of development to ensure quality.
- vi. In the case that a major requirement has been inaccurately assessed, a meeting with the major stakeholders will be initiated to resolve the issue. Incompatible smaller scale requirements will be addressed and resolved by the project lead.

6. Reuse Analysis

UberSoft has decided that they will be implementing various open source components in UberCMS. This decision will allow UberSoft to spend more time working on UberCMS' unique features instead of wasting resources developing components which are already readily available as open source.

The open source components chosen were picked with strict criteria in place. They must be easy to implement into our system, and they must be well established and tested to ensure that all the bugs have already been eliminated. By following these criteria it will ensure that we don't run into problems and we actually do save time implementing open source instead of writing the areas from scratch.

The two components that UberCMS will be using open source for are the message forum and the RTF document editor. We choose these areas because they were the high risk areas which could of taken away too much development time away from other areas.

The message forum software we are going to implement is called TribbyBoard. TribbyBoard was chosen due to it being a simple, easy to setup threaded message board under the GPL public license without many of the useless features on some of the other boards available.

The RTF editor that we have chosen is called htmlArea. HtmlArea is a free web based document editor written in Javascript. It's really easy to use and would allow users to fully edit their documents online which would make a great feature in UberCMS.

More information on TribbyBoard can be found at
<http://www.tribby.com/board/>

More information on htmlArea can be found at
<http://www.codeproject.com/jscript/htmlarea.asp?df=100&forumid=4685&exp=0&select=743075>

7. Approaches to Standard Functional Areas

This section will describe and explain the methods in which standard function areas of UberCMS will be built.

7.1 Database Organization

The database will contain information about Announcements, Calendar Events, Forum Posts, Projects, Files and Users.

These are a series of diagrams to illustrate the structure, content and organization of the database. PK refers to the primary key of the table.

Announcement	
PK	<u>A ID</u>
	Date Project ID

Event	
PK	<u>Event ID</u>
	Date Project ID

Project	
PK	<u>Project ID</u>
	Date

File	
PK	<u>File ID</u>
	Date Project ID Author Status Name

User	
PK	<u>User ID</u>
	Name Email Organization Password Permissions Project ID

Forum Post	
PK	<u>Post ID</u>
	Date Project ID Author

Each table contains a field entitled “Project ID” to associate each entity with a project. Every table except the User table includes a “Date” field to indicate the creation or last modification of the entity.

The “Author” field in the File table is used to store the name of the creator or the person that last modified the document. The “Status” field indicates whether a file is locked or unlocked. If a file is locked, it may be opened or downloaded but cannot be edited or uploaded until it has become unlocked.

The User table includes a field called “Organization” to store the name of the company or organization that the user belongs to. The “Password” field in the user table is checked every time that any user logs into the system. The “Permissions” field may include the values “Administrator”, which refers to a system administrator, “Project Leader”, which

refers to the leader of a specific project and “User”, which refers to any other user of the system.

Each Forum Post has a field called “Author” that contains the name of the user that created the post.

7.2 Data storage

All data that is not stored on the database will be stored on the Beddie server.

Non-database storage includes all files, templates, and the content of announcements, forum posts and calendar events. Content refers to the actual text. For example, the content of an announcement is the body or text which may be something like “The due date for the Requirements document is March 14 2006”.

7.3 Key Algorithms

For the first stage of delivery, the system will only include core functionality. Core functionality refers to the ability to logon, upload and download files, have an announcements page and possibly a contact list. Although the implementation will probably go past the first stage of delivery, the goal is to complete at least the core functionality and add other important functionality to it as the project progresses. Since the major concern is core functionality, no major algorithms will be used.

If there is time to implement all the stages of delivery, key algorithms may be considered after a certain amount of core and important functionality is implemented.

7.4 Memory Management

The server will have a directory structure with a directory for each project. In the directory for each project there will be subdirectories which may contain either files or subdirectories containing other subdirectories and/or files.

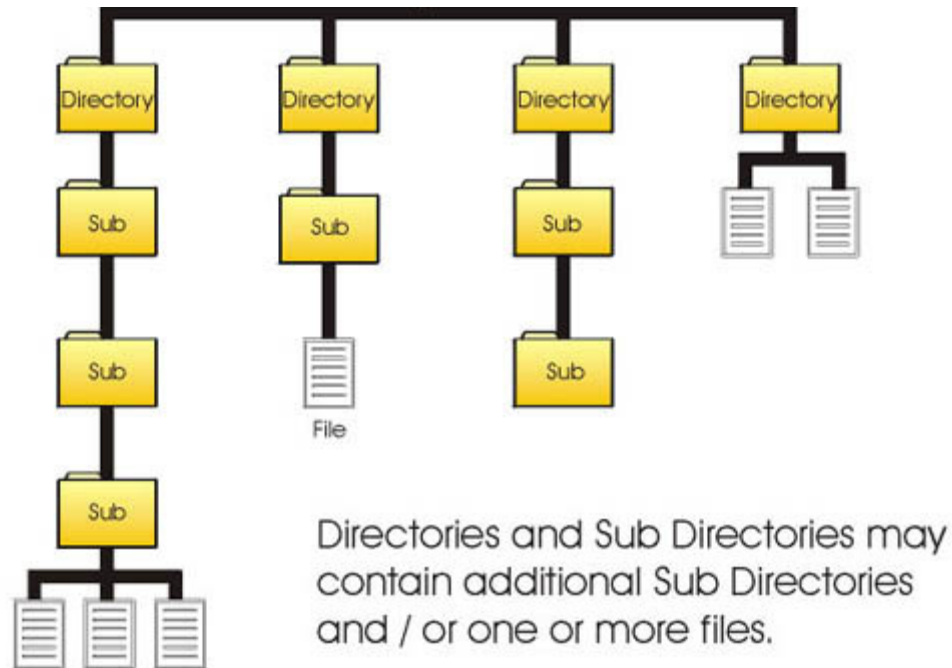


Image from: <http://www.sherwoods.com/Images/Misc/DIR2.jpg>

The database will contain information about all of the project data in tables. As mentioned above in the Database Organization section, there will be separate tables for announcement, event, project, user, file and forum post.

7.5 String Storage

Users will be informed of any errors that are caused as a direct response to an action they have taken and that are relevant to them. Error messages will be comprehensive and detailed, in other words the users will know exactly what caused the error and possibly how it can be fixed.

7.6 Concurrency and threads

The MySQL database that will be used can handle both threading and concurrency. Although the system will not use or allow multithreading, concurrency is a very important feature or concept to the system.

These are some examples of concurrency within the system:

File Access	-file may be opened or downloaded by multiple users at the same time -only one user can edit a file at a time
Forum Post	-multiple users may view a post at the same time
Announcements	-multiple users may view an announcement
Calendar Events	-multiple users may view a calendar event
Log In	-multiple users may log in at the same time

7.7 Security

There are several procedures to ensure system and user security.

Logging in: When a user is created, they are assigned a password. After the first login to the system, the user may change his or her password.

Permissions: Each user is assigned a certain status to indicate what permissions they have on the system. An “Administrator” is a system administrator and has access to everything on the system and can make any changes or additions that he or she wishes to make. A “Project Leader” is the leader of a project and can make changes within the project that they lead. A “User” can only access files in projects that they belong to. A User can download, open, edit and upload files within their designated project. A User may also make new posts on their project forum. A User may view any files, announcements, calendar events and forum posts within a project that they belong to. A User may also view the project’s contact list.

Locking Files: When a file is being edited by one user, it is automatically locked by the system. This means that no other user may edit the same file until it becomes unlocked. An administrator may unlock and lock files.

7.8 Localization

This version of UberCMS is targeted toward a Western English speaking audience. While a more localized approach would be ideal, the current project constraints do not allow for a variety of languages or compliance of the system with different world cultures.

It is possible for users of any country to use UberCMS, providing that the user can understand English.

7.9 Networking

A critical feature of any Content Management System is support for multiple users. This feature is important because it allows users to use the CMS at any time of the day without having to worry about whether other users are using the system at the same time.

UberCMS will support multiple users and concurrent actions. This means that multiple users may perform the same or similar action at the exact same time. An example of this is allowing multiple users to log in at the same time. Other examples of concurrent activities are presented in the “Concurrency and Threads” section of this document.

7.10 Portability

UberCMS is going to be a web-based system, and accordingly browser portability is the only portability concern or issue. This system will support all the popular browsers (i.e. Internet Explorer and Mozilla Firefox).

7.11 Programming Language

Many different programming languages and tools will be used to build and manipulate different aspects of the system.

Dreamweaver and HTML: will be used to construct the web interface.

PHP: will be used to implement the functionality of the system

MySQL: will be used to manipulate the database

7.12 Error Handling

All errors will be propagated up through the call chain to the interface, where errors will be reported.

If an error is caused by the user, the action that the user was trying to perform will not be performed. The user will be given an appropriate error message that will let them know what has gone wrong. The user will also be told how to “fix” the error – in other words how to complete the action in another way in which an error will not be caused.

If an error interrupts the action of another user that did not actually cause the error, an error message may be provided to let that user know that their action hasn’t gone through. Another way that the system may respond to this is by recovering from the error and making sure to redo the action that was being performed by this other user.

7.13 Documentation

UberSoft will provide documentation with UberCMS to assist the user. The documentation will include a visual breakdown of the different components and areas of the web interface by making use of screenshots. It will also include a section to explain, step by step, all actions that the user may perform. The generic documentation will be targeted toward the user that is neither a system administrator nor a project leader. There will be additional documentation for system administrators and project leaders to cover any areas that were not covered in the generic documentation.

8. Requirements Traceability & Testing

For quality assurance purposes, a spreadsheet will be used to track the progress of the product's requirements being met. Testing will be done concurrently to the development process, and the spreadsheet will be updated to indicate any problems, resolutions, and goals achieved. The spreadsheet will be cross-referenced with our SRS document. Below are the main areas to be tested.

Requirement #	Description	Associated Subsystem
1.1	Store old version of files.	File System
1.2	Display all the files that a user has access to.	File System, Database
1.3	Allow users to download accessible files.	File System
1.4	Allow users to edit accessible files	File System
1.5	Only allow one user to edit a particular file at a time.	File System
1.6	When a user chooses to edit a file, it will be opened in an rtf editor.	File System
1.7	Administrator may unlock a file.	Database
1.8	Administrator may lock a file.	Database
1.9	Allow users to upload accessible files that are not on lockdown.	File System, Database
1.10	Allow users to save files.	File System
1.11	Users may create new files.	File System
1.12	Allow users to open accessible files.	File System
1.13	There are templates for the user to use.	Database
1.14	Time stamp files upon creation.	File System
1.15	Time stamp files when edited	File System
1.16	Maintain directory structure for file browsing.	File System
2.1	Allow directory creation.	File System

2.2	Allow directory deletion.	File System
2.3	Allow renaming of directories.	File System
2.4	Allow moving of files from one directory to another.	File System
2.5	System must allow user to login/logout	Database
2.6	System must have different levels of user access rights: worker, project leader, administration	Database
2.7	Must allow only certain users access to each project	Database
2.8	Must allow Worker write access to all files within their designated projects	Database
2.9	Must allow Project leader to control who has access to their project.	Database
2.10	Must allow Project leader and Administrator who can set or change deadlines.	Database
2.11	Must allow Administrator to be the only person who can create or remove projects, and create or remove users.	Database
2.12	All important deadlines must be able to be seen in a user's calendar	Database
3.2	Daily, monthly and total project timelines of the project may be viewed in a user's calendar	Database
4.1	Allow user to post message on forum.	Database
4.2	Allow user to reply to posts on forum.	Database
4.3	Allow user to edit own posts.	Database
4.4	Administrator can delete posts.	Database
4.5	Allow user to create new thread.	Database
4.6	Administrator or Project Leader can create announcements.	Database
4.7	Recent announcements displayed on announcements page.	Database
4.8	Announcements older than 14 days are in announcements archive.	Database
4.9	Users can view announcements on announcement page.	Database
4.10	Users can view archived announcements.	Database
5.1	Each project has a contact list.	Database

5.2	Contact lists relevant contact information for members of project.	Database
5.3	Users can email people on contact list.	Database
5.4	Users may email everyone on contact list.	Database

9. Support for Staged Delivery

9.1 Procedure for Release

Our software product will be released in 4 stages. Stage 1 will consist of the critical components needed to produce a basic skeleton for future releases. This stage will not be released to the public, and will be used for in-house testing and development. The bulk of our product will be released in the second stage. Stage 2 will consist of basic file control and manipulation, security, documentation, and archiving of files. Stage 3 includes enhanced documentation, messaging to other project members, forum, and creation of new RTF files. Stage 4 consists of the calendar, project timelines, and all other seen and unforeseen enhancements and modifications for the future.

9.2 Stage 1 – Critical Components

Release Version 0.1 will implement the following features:

Feature	Notes	Requirement Number
Web-based interface	- Main start-up splash screen and the style format of basic web pages constructed including login, visible projects, menus	N/A
Database connectivity	- Server (Beddie) is configured and operational, connections to and from server will be confirmed	N/A
Database configuration	- Database tables and fields constructed for preliminary testing using MySQL	N/A

9.3 Stage 2 – File control and manipulation

Release version 0.2 will implement the following features:

Feature	Notes	Requirement Number
Login	- User able to login to the system and be able to view and access different projects depending on identification and privileges	1.2 2.5 2.6 2.7
File access	- Users able to open, download, edit, save, upload files of different varieties - User identification, modification data appended to files	1.3 1.4 1.6 1.10 1.11 1.12 1.14 1.15 2.8
Security	- Administrator account created, user accounts created, accessibilities determined, locking of files	1.5 1.6 1.7 1.8 2.9 2.11
Directory manipulation	- Project organization of files to be managed by users	2.1- 2.4
Templates	- Users may create files based on a template	1.13
Archiving	- Archived files stored with modification history, version numbers - Archives retrievable	1.1 1.14
User Manual	- Deliverable for users on usage guidelines and features	N/A

9.4 Stage 3 - Messaging, RTF Creation, Enhanced Documentation

Release version 0.5 will implement the following features:

Feature	Notes	Requirement Number
Announcements	- Administrator and Project lead can make new announcements - Announcements viewable by all intended	4.6-4.9 4.10
Forum	- Users can make, edit, view posts	4.1-4.3 4.5
RTF editor	- In-house editor available to make new files or modify already existing ones in project	1.6
Contact list	- Project members' contact information displayable, menu option available	5.1 5.2
Email	- Users can email other project members	5.3 5.4
User manual	- Updated user manual from release version 0.2	N/A

9.5 Stage 4 – Future Enhancements

Release Version 0.9 will implement the following features:

Feature	Notes	Requirement Number
Calendar	- User calendar feature available via menu, alerts	2.12 3.2
Project timelines	- Deadlines can be set by Project Lead	3.3
Others	- Unforeseen modifications needed because of change cases or requirements re-specification during our iterative development process	N/A
User manual	- Updated user manual from release version 0.5	N/A

10. Conclusion

The architecture specified in this document will allow Ubersoft to develop UberCMS without running into any issues. All the high risk areas are already taken care of by using open source components and the lower risk areas can be easily implemented by following the diagrams and structure we have outlined. They detail almost all aspects of UberCMS including how it will run and how it will be created. The functional requirements will be tested during implementation using white box testing. The non-functional requirements are mostly usability based and will be met by fully testing our system using non-project related individuals and ensuring that it is possible for these users to learn how to operate the system in less than an hour.

Appendix A

Glossary

Beddie – the server that will be used to store non-database information for the system.

Component - An independent section of software.

Concurrency – when one or more operations or functions are performed at the same time, or overlap in time.

Directory – the same as a file folder.

GPL Public License - A license which grants any user the right to copy, modify, and redistribute programs and source code.

Multithreading – when a system or computer executes different threads of a program at the same time.

Open Source - Software that comes with its source code.

RTF document - A type of document using the ASCII character set which normally end in the .rtf extension.

Subdirectory – a directory within a directory.

Threading – when a program is broken down into many sections or tasks that can run independently of one another.

UML –the standard for modeling software artifacts.

User – relates to any level of user of the system, i.e.: can be administrator, project leader or other user.

Worker – see User.

XML - Extensible Markup Language. A flexible way to create common information formats and share both the format and the data on the World Wide Web, intranets, and elsewhere.