CERTIFICATION OF APPROVAL

UTP WEB DESKTOP ENVIRONMENT

By

Mohd Zafar Bin Ramli

A project dissertation submitted to the
Information & Communication Technology Programme
Universiti Teknologi PETRONAS
in partial fulfillment of the requirement for the
BACHELOR OF TECHNOLOGY (Hons)
(INFORMATION & COMMUNICATION TECHNOLOGY)

Approved by,

_____

(Mr. Mohammad Noor Ibrahim)

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK
January 2007

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

MOHD ZAFAR BIN RAMLI

ABSTRACT

This report describes about the implementation of UTP Web Desktop Environment using Asynchronous JavaScript and XML (AJAX) as main programming languages. The UTP Web Desktop Environment is an online 'desktop' workspace for student where it contains multiple applications that can be accessed simultaneously within a single web browser. The system may promote a new way of experiencing web applications where students are given a bunch of ways to manipulate the system. The objective of the development of the system is to provide a web-based desktop environment that allows user to interact with the desktop workspace as same as user's computer operating system's desktop. Due to some limitation of conventional web applications where most of the content of it is static and dull, it is also the aim of this research to explore the feasibility of using AJAX as the main programming language since it make applications more responsive, interactive, and customizable. To carry out this study, some researches have been made by comparing the requirement of this system with a similar system, WebOS which leads to the objectives of this application. Based on the flow of RAD, the prototypes are developed along with the new ideas of designing it specifically for student's interest. Some researches have also been made about RSS and AJAX's components, requirements, and implementation to distinguish the advantages of using them rather than using other programming languages. The application is driven by EyeOS MicroServer which responsible for managing web server and AJAX compiler. By implementing this project for UTP student, the author can conclude that it will provides students with a cutting edge systems that never been applied before where students may find it very helpful and interesting to organize their live and work.

ACKNOWLEDGEMENT

I would like to express my gratitude first and foremost to Almighty Allah S.W.T for giving me this opportunity and success in doing Final Year Project entitled, UTP Web Desktop Environment. Special thanks to my supervisor, Mr. Mohammad Noor Ibrahim, for taking me under his wing and patiently guiding me through it all. Thanks also to the other ICT/BIS lecturer for your guidance in completing this course especially to Mr. Hilmi Hasan. Thank you to my beloved parents for giving me unlimited support during my project research and development. To my friends and course mates who always willing to share their knowledge and encourage, I thank you. Last but not least, I would also like to express a special thank to those who have directly or indirectly contributed in doing this Final Year Project.

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND OF STUDY

As the Internet has become more mature, rich applications featuring responsive user interfaces and interactive capabilities have become increasingly popular. The capabilities represent a way to make programs easier to use and more functional, thus enhancing the user experience. The development of such interactive applications is become essential to students where internet is one of the most used medium for searching online sources and communications. But some existing applications developed for student is not sufficient enough to fulfill their curiosity for interactivity.

Student may find it hard to perform their daily tasks, because of the limitation of the conventional web applications. For example, whenever a student wants to use two applications simultaneously, the student must have to open two web browsers to do so. Thus, student may require an online workspace where it provides various applications such as messaging services, bulletin board, virtual hard-disk, word processor, and media players; in a single web page; which each application can be accessed simultaneously on a single web browser.

With this "UTP Web Desktop Environment" system, it may promotes a new style of experiencing an online workspace where students have a lot of freedom to interact and manipulate with the applications. This UTP Web Desktop Environment System will give users an opportunity to have their own online virtual 'desktop' workspace that they can access it anywhere and manage their own desktop the way they want it to be and expose

users to a new style of organizing their life and work. Users may organize their calendar, working on their project using provided word processing application and having fun with media players just by login to their account at places like homes, hostels, labs, or any places that provided internet connection.

Imagine being able to sit down at any ordinary computer in an Internet cafe or public library, start up the resident web browser and access your personal desktop – along with your applications, messages, and information – with a click of your mouse button. Then, when you're done, you simply close the web browser to have all of the documents you've just worked on safely stored on your remote server, leaving nothing on the computer you've just been working on.

Research and study will cover from the problem identification, objectives of the project, scope of study, literature review and methodology of the project to come out a report. This research may also cover multiple programming languages such as JavaScript, PHP, and AJAX that can be used in developing the website. Based on the studies, smaller part of working module will be identified to be developed and to be combined as a prototype. The prototype of the product will be a version 1.0 of the web-based system, which resides in a Windows-operated server. The system is developed in AJAX (Asynchronous JavaScript and XML) languages with the use of Macromedia ™ Dreamweaver as the main development tool and Eyeos MiniServer as the web server.

1.2 PROBLEM STATEMENT

1.2.1 PROBLEM IDENTIFICATION

The following are identified problems from the study:

1) **No complete and convenient Web-based Operating System that facilitates web-based desktop for UTP student**

   Currently, there is no complete and convenient Web Desktop Environment that provides a desktop environment for users to interact and use the provided services such as word processing, bulletin board, messaging, etc.

2) **Conventional web-based applications are not providing user with the freedom to interact and manipulate the applications**

   Most of conventional web-based applications are not giving enough freedom to a user to interact and manipulate the data such as drag and drop, open multiple applications at a time, minimize, maximize, and automatically refresh the web-browser.

3) **The conventional web-based applications are disruptive and lowers productivity**

   Conventional web-based applications require a user to submit a request, wait for the server to response, and then wait for the web-browser to update the data by reloading the entire page. Thus, this pattern is disruptive and lowers productivity of a system.

4) **Most of those web-based applications cannot interact without been asked.**

   Mostly, the conventional web-based applications are only responding to a request made by a user. This web-based application implement a push and pull technology that allows the system to work by itself without been asked.

1.2.2 SIGNIFICANT OF THE PROJECT

The implementation of the project promises a new way for a web-based application to provide flexibility to users without threatening the network capabilities. The system will provide a new look of a web-based application by removing static contents and allow user to manipulate the contents without being restricted by website itself. Many said that by implementing this kind of applications through internet may burden network capabilities to send and receive data simultaneously. But this application will be developed using AJAX (Asynchronous JavaScript and XML) languages that dynamically reply the user requests with the result without resending the whole pages. Thus, only small portion of data is send and receive simultaneously at a time.

1.3 OBJECTIVES AND SCOPE OF STUDY

1.3.1 OBJECTIVES OF THE PROJECT

Objectives focus on the goal of this project and purpose of completing this project. The followings are the identified objectives of this project.

1) To perform a research about existing web-based operating system's capabilities in term of usability, functionality, reliability, and network loads. The research will also cover the languages that should be used to make this application as flexible as it can be.

2) To provide a web-based desktop environment that allows user to interact with the desktop as same as user's computer operating system's desktop. Users can drag icons anywhere he want it and click on it to start the application, add shortcuts, remove unwanted files by drag the file to the recycle bin, open multiple applications at a time,

maximize and minimize them, and move or resize the applications. These flexibilities are not provided in most of conventional web-based applications.

3) To give users the sense they were using a desktop application instead of an Internet-based application. It is a challenging to create such application that some big organizations recently adapt it to their applications such as Google Gmail, Google Maps, and Yahoo! Mail Beta.

1.3.2 THE RELEVANCY OF THE PROJECT

The relevancy of the project will be the importance and significance of this project to solve the problems as stated in the problem statement. The importance or benefits that a user can enjoy from implementing this system are:

1) Users can achieve mobility where they can access to his personal web-based operating system anywhere at any ordinary computer and do their work on the web-based operating system and save it when they done, then close the web browser to have all of the documents you've just worked on safely stored on your remote server, leaving nothing on the computer they've just been working on.

2) The system gives users with the freedom to interact within his personal web-based operating system and manipulate the data such as drag and drop, open multiple applications at a time, minimize, maximize, and move or resize them.

3) This system is not using the conventional request-wait-response-wait pattern. This system only load all its components once at client web-browser, then any interactions within the system only involve in small portions of data transfers. Thus, this will reduce loads within the networks.

4) This system works dynamically without having users to request every time they want to use it. The system pushes the data automatically to the user without waiting the user to request it.

1.3.3 SCOPE OF STUDY

This project focuses on providing a web-based interface that consists of many functionalities to make it flexible allowing drag and drop, maximize, minimize, move, and resize applications and. All the main functionalities of this system are broken into smaller modules, so that they are easy to develop and managed. A prototype will be developed based on the modules identified.

This project also focus on developing applications that are suitable for UTP student such as virtual hard disk, personal messages between users, bulletin board for user to post updates, announcements, and news, etc. This project will be developed within the time frame of 12 weeks duration. The allocated time frame should be enough to carry out the necessary research as well as development work of the prototype of the system according to the identified modules. The Gantt chart is developed in order to plan and to oversee the entire project progress so that the project is delivered in time.

CHAPTER 2

LITERATURE REVIEW

2.1 WEB-BASED OPERATING SYSTEM (WebOS)

Initially, operating systems were developed to provide a set of common system services such as managing I/O, communication between devices, storage, to simplify application programming. With the advancement in developing operating system, world is now looking forward into an operating system that provides a workspace for wide area applications. The introduction of local area networks in the 80's expended this role even further [11]. A goal of network operating system was to make remote resources over the LAN as easy to use as local resources, in the process simplifying the development of distributed applications. With the analogy of the system, it is time to provide a common set of services for wide area applications using LAN, WAN, and Internet as the medium of communications [6].

Although the World Wide Web has made geographically read-only data easy to use, geographically distributed computing resources are not [2]. The result is that wide area applications that require access to remote CPU cycles, memory, or disk must be programmed in an ad hoc application-specific manner. For example, many popular services such as Digital's Alta Vista or Netscape's download page, are geographically replicated to improve bandwidth, reduce latency, and improve availability – no single connection into the internet can supports tens of millions of users at a time[1]. This replication is managed by hand on both the server and the client side – users are forced to do manual polling between essential equivalent services. This situation will get worse since it currently predicted the number of internet users will increase by an order of

magnitude to over 100 million in less than 5 years. To address these problems, the author introduces a web-based operating system.

2.1.1 Overview

The web-based Operating System framework enables a new paradigm for Internet services. Instead of being fixed to a single location, services can dynamically push parts of their responsibilities out onto Internet computing resources, and transfer all the way to the client. This may provide a number of advantages, including [13]:

i) better end-to-end availability (service-specific extensions running in the client mask Internet or service failures),
ii) better cost-performance (by dynamically moving information closer to client, network latency, congestion, and cost can all be reduced while maintaining server control), and
iii) better burst behavior (by dynamically recruiting resources to handle spikes in demand).

The development of this web-based operating system will establish an extensible mechanism for running service-specific functionality on client machines to show that this mechanism allows more flexible implementation of name resolution, load balancing, and fault tolerance [1]. This development will also find out that it may simplifies the implementation of a number of wide area applications since this operating system will be a standard platform for those applications to run. Next, we will find out by implement the coherently caching program through the file system for this operating system will speed up the performance of applications which must repeatedly execute programs with common inputs.

2.1.2 Components

There will be several major components needed to make the operating system works.

i) **Resource Discovery**

Many wide area services are geographically distributed. To provide the best overall performance, a client application must be able to dynamically locate the server to deliver the highest quality of service. This can be done by mapping a service name to multiple servers, an algorithm for load balancing between servers, and maintaining enough state to perform fail-over if a server becomes unavailable [1].

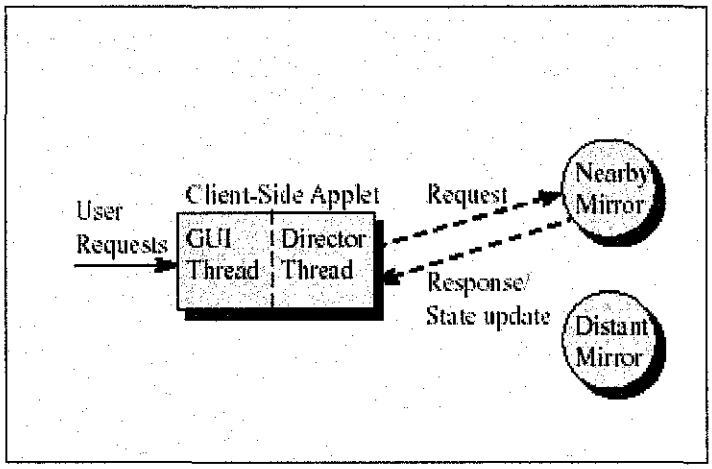

Figure 1: WebOS Architecture [1]

The diagram above shows that two cooperating threads of webOS make up the operating system architecture. The GUI thread presents the service interface and passes user request to the Director Thread. The Director is responsible for picking a service provider likely to provide best service to the user. The decision is made in a service-specific manner. In this case, the nearest mirror site is chosen.

The architecture of the resource discovery is summarized in figure 1. To achieve the objective of resource discovery management, we need to consider

2 approaches. First, a service name must be mapped onto the replicated service representatives. Next, a load balancing decision must be made to determine which server is able to deliver the best performance.

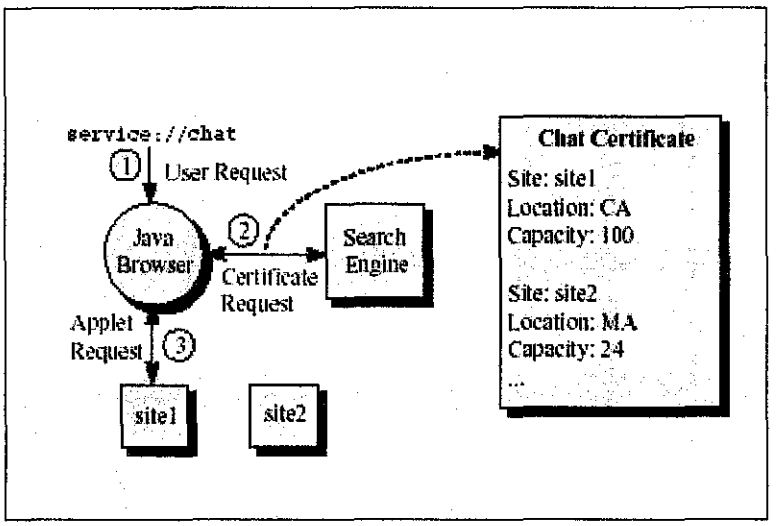

Figure 2: Bootstrapping Applet Retrieval [1]

Figure 2 summarized the approach to overcome problems with the resource discovery such as central bottleneck, a single point of failure, and doubling latency for small request. A meta-applet runs in a java enabled browser and is responsible for bootstrapping accessing to the services. A new service name space is introduced, allowing users to make request of the form of *service://URL*; we need to leverage the URL name space to simplify integration with the existing browsers. The meta-applet translates these names into request to a well known and highly available Internet name service to fetch a *service certificate* – the list of servers capable of providing the service [1].

The implementation of the UTP Web Desktop Environment will developed on a single server during prototyping phase. The system will be geographically distributed among servers later to reduce data lost and latency problems when multiple users accessed it simultaneously. The algorithm of selecting server will be defined based on least accessed server at the time.

ii) **Wide Area File System**

To support replication and wide-scale sharing, we need to implement a cache coherent wide area file system. The system will extends to wide area applications to run in a secure HTTP name space at the same interface, caching, and performance of existing distributed file systems. A fundamental difference between this File System and existing Internet Naming and caching proposals is that this File System is designed to be used by programs, not just by individuals accessing infrequent updated data [13]. We also may implement Internet Push Technology using IP multicast [9].

However, the implementation of UTP Web Desktop Environment version 1.0 will not be replicated and distributed as said above because the having multiple servers running at a time is not cost effective as this system is only an alternative way of experiencing a new way of managing student's work and life. But it will be a good idea to provide a good overall performance later.

iii) **Security and Authentication**

To support security and authentication of the user, the system will provide a key enabling feature that offers the control of capabilities to execute remote processes on behalf of users. We need to provide security at many levels. First, two principals communicating at the link layer believe one another's identity and trust that the data cannot be compromised by a third party [6]. Next, principals are able to have fine-grained control over which capabilities are transferred to remote operating system process on their behalf. Finally, this system needs to provide an interface for registering users and for specifying access rights to individual system resources [6].

For the UTP Web Desktop Environment, this feature is crucial to distinguish user's individual setting and files of each account. It also provides a privacy statement for user to keep their personal files and information.

iv) **Process Control**

The main responsibility of this system is to execute a process on a remote node should be as simple as corresponding local operation [1]. The underlying system is responsible for authenticating the identity of the requester and determining the proper access rights are held. Precautions must be taken to ensure that the process does not violate local system integrity and that it does not consume more resources than allocated to it by the local system administrator [2].

The implementation of the UTP Web Desktop Environment will ensures any actions within the system will not jeopardize user's local machines integrity and resources as it only consume a small resource of client-side machines.

2.2 AJAX: ASYNCHRONOUS JAVASCRIPT AND XML

Conventional browser-based web application require the user to submit a request to the server, wait for the server to process the request and generate a response, and then wait for the browser to update the interface with the result. This request-wait-response-wait pattern is extremely disruptive and lower productivity. High network latency and interface complexity, and slower server responsiveness can further impair the user experience, resulting in decrease customer satisfaction, shorter and less frequent website visits, and ultimately reduced revenue to we applications that implementing e-business. To overcome these problems, world is now been introduced to Asynchronous JavaScript and XML, also known as AJAX programming.

2.2.1 Overview

Asynchronous JavaScript and XML is a standards-based programming technique designed to make web-based applications more responsive, interactive, and customizable – in short, to recreate the seamless user experience of most other desktop applications. There five characteristics of application built using Ajax [3]:

i) a user interface constructed with open standards such as the Dynamic Hypertext Markup Language and cascading stylesheets (CSS);
ii) a dynamic, interactive user experience enabled by the document object model;
iii) data exchange and transformation using the Extensible Markup Language (XML) and Extensible Stylesheets Language Transformations;
iv) asynchronous client/server communication via XMLHttpRequest; and
v) JavaScript as the lingua franca joining all the components together.

2.2.2 Advantages of AJAX

Ajax offers many advantages over conventional approaches to web application development. The primary advantages of Ajax-style web applications are less waiting and more control for the user [3]. Ajax accomplishes this by

i) eliminating full-page post-backs in favor of smaller, incremental in-place updates;
ii) leveraging the client machine's processing power and temporal proximity by making the web browser responsible for more aspects of the application execution; and
iii) exploiting modern web browsers' rich graphic capabilities – transparency, shading, animation, Z-ordering, compositing, and so on – to add more glitz and interactivity to the presentation of information.

2.2.3 AJAX Component Technologies

Most of Ajax's component web technologies were developed and standardized during past 10 years [4]. These technologies have improved recently, making them more suitable for enterprise use.

2.2.3.1 Dynamic HTML

Ajax applications take advantage of dynamic HTML (DHTML), which consists of HTML, cascading stylesheets, and JavaScript glued together with the document object model. The technology describes HTML extensions that designers can use to develop dynamic web pages that are more animated than those using previous HTML versions. For example, when a cursor passes over a DHTML page, a color might change or text might get bigger. Also a user can drag and drop images to different places.

2.2.3.2 Extensive Markup Language (XML)

Ajax uses XML to encode data for transfer between a server and a browser or client application. The W3G started work on XML in 1996 to enable cross-platform data interoperability over the Internet. The consortium approved the standard's first version in 1998. XML is a markup meta-language that can define a set of languages for use with structured data in online documents.

2.2.3.3 Cascading Stylesheets (CSS)

Cascading Stylesheets, better known as CSS gives website developers and users more control over how browser display pages. Developers use CSS to create stylesheets that

define how different page elements, such as headers and links, appear. Multiple stylesheets can apply to the same web page.

2.2.3.4 Document Object Model (DOM)

The Document Object Model is a programming interface that lets developers create and modify HTML and XML documents as sets of program objects, which makes it easier to design web pages that users can manipulate. The DOM defines the attributes associated with each object, as well as the way in which users can interact with objects. DHTML works with DOM to dynamically change the appearance of web pages. Working with DOM makes Ajax applications particularly responsive for users.

2.2.3.5 JavaScript

Released in 1995 by Netscape and Sun, JavaScript interacts with HTML code and makes web pages and Ajax applications more active. For example, the technology can cause a linked page to appear automatically in a popup window or let a mouse rollover change text or images. Developers can embed JavaScript, which is openly and freely available, in HTML pages. Ajax uses asynchronous JavaScript, which an HTML page can use to make calls asynchronously to the server from which it was loaded to fetch XML documents. This capability lets an application make a server call, retrieve new data, and simultaneously update the web page without having to reload all the contents, all while the user continues interacting with the program. Because JavaScript is a cross-platform scripting language, Ajax applications require no plug-ins, unlike Macromedia Flash and other proprietary web application technologies.

2.2.3.6 XMLHttpRequest

Systems can use JavaScript-based XMLHttpRequest objects to make HTTP requests and receive responses quickly and in the background, without the user experiencing any visual interruptions. Thus, web pages can get new information from servers instantly without having to completely reload. For example, users of an application with XMLHttpRequest objects could type in a centigrade amount in one box of a temperature-conversion application and have the Fahrenheit amount appear instantly in another box. Various browsers such as recent versions of Internet Explorer, Mozilla Firefox, Netscape, and Apple's Safari work with XMLHttpRequest [15].

2.2.4 The Mechanism of AJAX

In the classic web application model, user actions trigger an HTTP request to a web server, which processes the request and returns an HTML page to the client. This makes technical sense but doesn't always provide a great user experience because, for example, it limits interactivity and require web pages to reload for every piece of new data. Ajax applications create a JavaScript-based engine that runs on a browser. Instead of loading a traditional web page, the browser load the engine, which then displays then requested materials, as Figure 3 shows. The engine intercepts user inputs and handles many interactions, such as simple data validation on the client side. If the engine needs more data, it requests the material from the server in the background without locking up the user interface. Thus, the engine lets user interact with an application independently of server communication, reducing server response wait times.

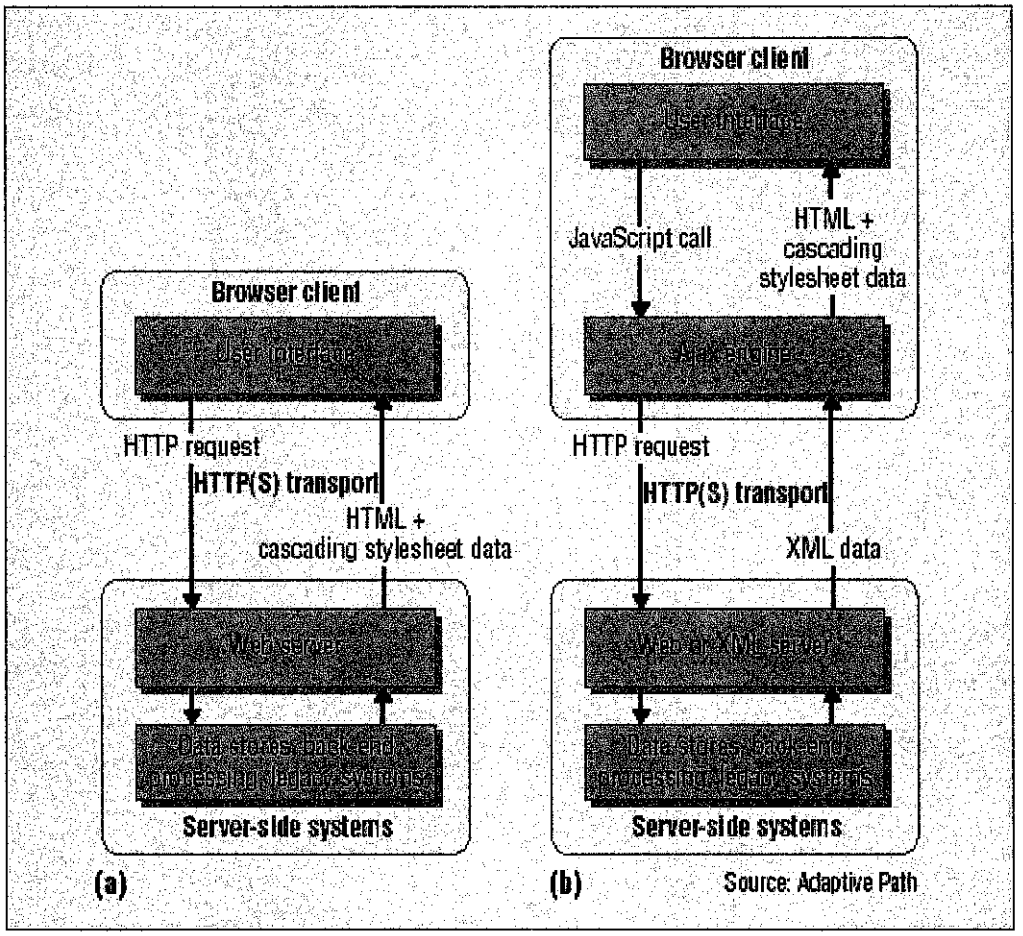

Figure 3: A Comparison between Conventional Web Application and AJAX Web Application [4]

From the figure above, In (a) a conventional web application, user actions trigger an HTTP request to a web server, which processes the request and returns an HTML page to the client. Additional requests lock up the application until the system update the page. (b) Ajax applications create a JavaScript-based engine that runs on a browser. The engine intercepts user inputs, display requested material, and handles many interactions on the client side. If the engine needs more data, it requests material from the server in the background, while letting the user continue to interact with the application.

Thus, Ajax is selected as the best programming language for developing UTP Web Desktop Environment as it allows integration among system files to provide more responsive, interactive, and customizable web applications.

2.3 ARCHITECTURE OF PULL AND PUSH TECHNOLOGY

Information sharing in the Internet is currently based on two information retrieval paradigms, namely pull technology and push technology. According to the pull technology, data download is triggered by an explicit and intentional action of the user of a web browser [8]. The main advantage of such an approach is that the user is drives his custom information ride by surfing at will on the hyperlinks. But the main drawback is the lack of any connection between the process of content creation and content access. Hence timely access to web content is hardly achieved unless the user himself frequently polls the website likely to generate valuable information. According to push technology, users express their interest in receiving information related to subjects of interest by subscribing to an information delivery channel. On the server side, content packager delivers data as soon as they are available by publishing them on the proper channels and all subscribers of the channel receive the same data [9]. But the main drawback of the push approach is that a user may then be flooded by the unsolicited delivery of data whose value to the user is far from being guaranteed.

2.3.1 Push Technology

Push technology, also called server push or webcasting, describes an Internet-based content delivery system where information is delivered from a central server to a client computer based upon a predefined set of request parameters outlined by the client computer. A client computer such as a desktop home user would subscribe to various information topics provided by a content provider and as that content is created by the content provider, such information is "pushed" or delivered across the Internet to the desktop home user and displayed on that user's computer. Push Technology differs from normal World Wide Web usage, where a user has to request a Web Site through a web browser [10].

E-mail is the classic Internet push media; however, this depends on the configuration used. If the messages are stored on a server and not automatically pushed to the client, then it is not technically push media. Instant messaging epitomizes push media. Messages and files are pushed to the user as soon as they are sent to the messaging service. Most web feeds, such as RSS, appear to be push media, but technically are pulled by the user. We will discuss about the RSS on the next chapter.

2.3.2 Rich Site Summary (RSS)

RSS is a format for syndicating content and metadata over the Internet. It is commonly used to share headlines and links to news articles. With news articles, the actual article isn't usually shared, but metadata about the article is; this metadata can include a headline, a URL, or a summary. RSS is an important tool for publishers because feeds can be used to syndicate content, and to integrate third-party content into your site.

RSS is a dialect of XML. All RSS files must conform to the XML 1.0 specification, as published on the World Wide Web Consortium (W3C) Web site [8].

Here's a typical example of how RSS is used:

- A publisher has some content that they want to publicize.
- They create an RSS channel for their content.
- In this channel, they include items for Web pages they want to promote.
- This channel can be read by remote applications, and converted to headlines and links. These links can be incorporated into new Web pages, or read in dedicated readers.
- People see the links on various sites, click on them, and go to the original publisher's site.

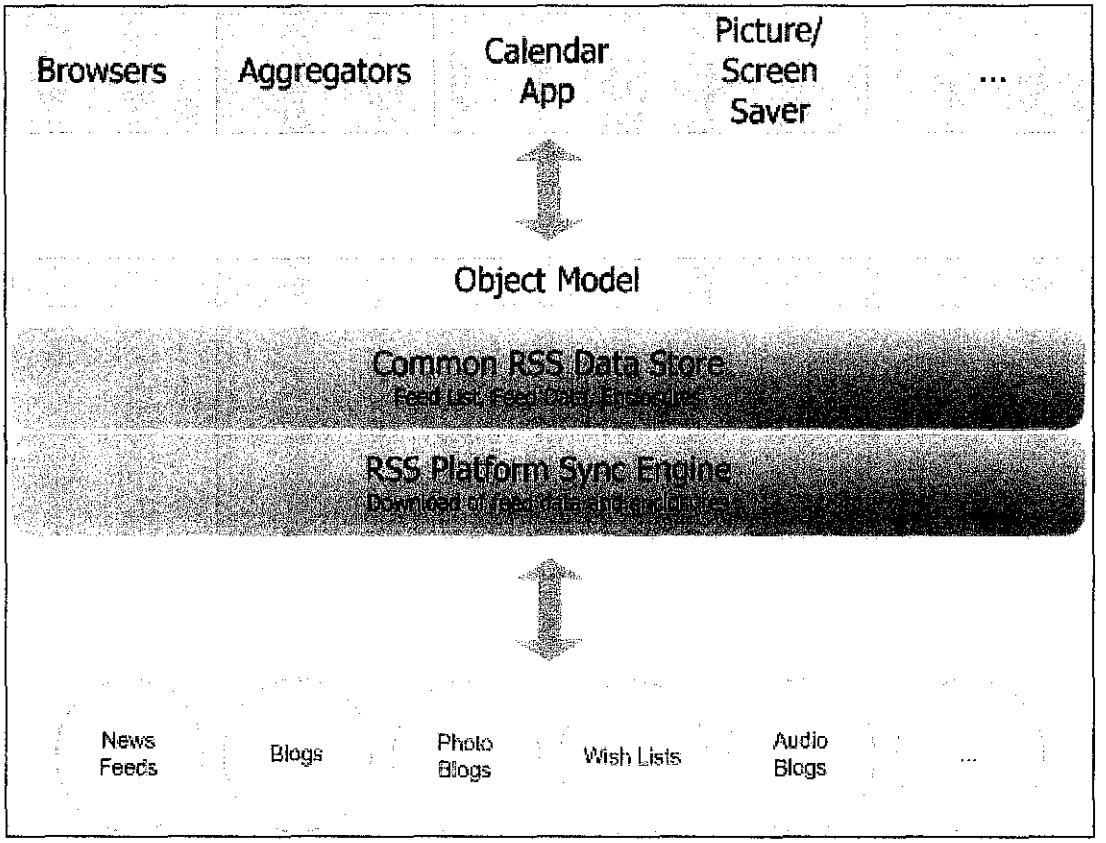

Figure 4: RSS Architecture [10]

While headline syndication is the most common use for RSS, it is also used for many other purposes. RSS is a very popular format in the weblog community. It's also used for photo diaries, classified ad listings, recipes, reviews, and for tracking the status of software packages.

RSS feeds are used in the world of e-commerce as a way of delivering information. For example, Amazon provides custom news feeds based on its Web services platform. This lets you track top books in your news reader, or include information on your Web site about related books for sale at Amazon.

RSS has grown tremendously in popularity in the last few years. Syndic8.com maintains an index of RSS channels, and its list of feeds has grown by about 1400% in two years. Yahoo news, the BBC, Slashdot, LockerGnome, Amazon, CNN, Wired,

Rolling Stone, and Apple Computer are among the many popular sources of RSS feeds [10].

2.3.3 RSS Compatibility Issues

For the most part, later versions in each branch are backward-compatible with earlier versions (aside from non-conformant RDF syntax in 0.90), and both versions include properly documented extension mechanisms using XML Namespaces, either directly (in the 2.* branch) or through RDF (in the 1.* branch). Most syndication software supports both branches. Mark Pilgrim's article "The Myth of RSS Compatibility" discusses RSS version compatibility in more detail [8].

The extension mechanisms make it possible for each branch to track innovations in the other. For example, the RSS 2.* branch was the first to support enclosures, making it the current leading choice for podcasting, and as of mid-2005 is the format supported for that use by iTunes and other podcasting software; however, an enclosure extension is now available for the RSS 1.* branch, mod_enclosure. Likewise, the RSS 2.* core specification does not support providing full-text in addition to a synopsis, but the RSS 1.* markup can be (and often is) used as an extension. There are also several common outside extension packages available, including a new proposal from Microsoft for use in Internet Explorer 7 [8].

The most serious compatibility problem is with HTML markup. Userland's RSS reader—generally considered as the reference implementation—did not originally filter out HTML markup from feeds. As a result, publishers began placing HTML markup into the titles and descriptions of items in their RSS feeds. This behavior has become widely expected of readers, to the point of becoming a de facto standard, though there is still some inconsistency in how software handles this markup, particularly in titles. The RSS 2.0 specification was later updated to include examples of entity-encoded HTML, however all prior plain text usages remain valid [9].

CHAPTER 3

METHODOLOGY / PROJECT WORK

3.1 METHODOLOGY

Throughout this project, Rapid Application Development (RAD) is chosen as the development methodology. RAD is a programming system that enables programmers to quickly build working programs. Historically, RAD systems have tended to emphasize reducing development time, sometimes at the expense of generating efficient executable code. RAD is a methodology for compressing the analysis, design, build, and test phases into a series of short, iterative development cycles. This has a number of distinct advantages over the traditional sequential development model. Iteration allows for effectiveness and self-correction.

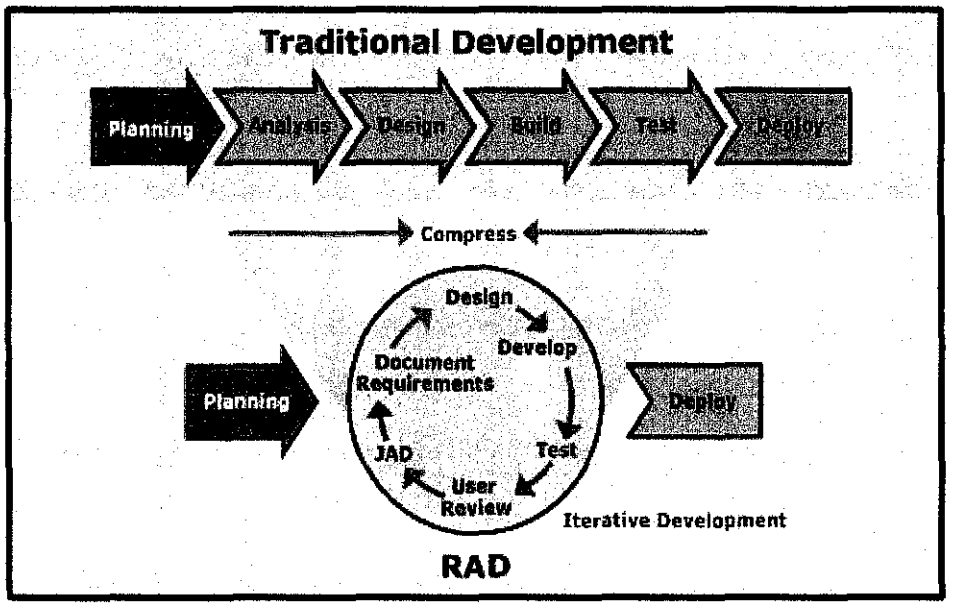

Figure 5: Rapid Application Development Processes

3.2 PROJECT WORK

3.2.1 Planning and Analysis

In this first phase, brainstorming and problem identification will be done. The clear objectives have been derived from the problem that had been identified. Multiple sources was tested and researched. Literature reviews were done. In this project, the problem statement and also the scope of study had been clearly stated in Chapter 1. The objectives had also been stated within the same chapter. And during this phase, the project timeline and milestone had also been develop according to the duration that had been stated out by the Final Year Project Committee.

3.1.2 Design

In this design phase, the initial concept and diagrams of the whole UTP Web Desktop Environment have been drawn. The design phase will correspond with the user feedback on each module that will be developed throughout the procedure. The design phase is the first phase of the iterative phase of the RAD methodology. During this phase, the main module in the system will be identified and some modification is executed according to the requirement of the system.

The design phase will be run through out the project duration, because of the nature of this iterative methodology. During this phase, the tools that are needed to be use in developing the system were identified. The best tools were selected in order to ease the development. The tools are expected to help cut the development time, while providing minimal cost.

After multiple entities for the system are identified, the author has come out with the use-case diagram and the data flow diagram to drive this project ahead and start

developing the prototype. The use-case diagram of the system is shown below as Figure 6 and the data flow diagram is shown in Figure 7.

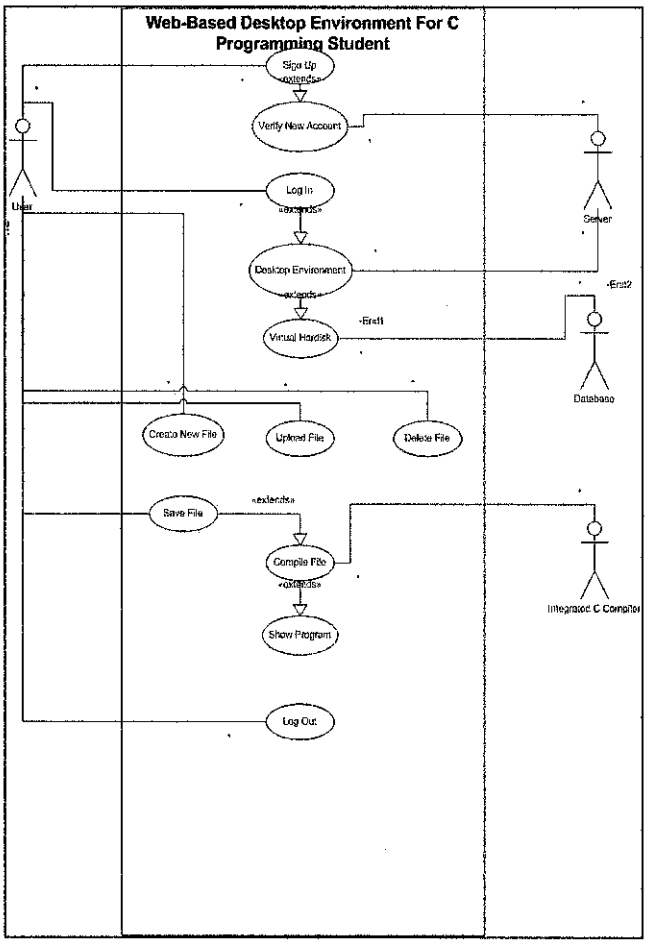

Figure 6: The Use-Case Diagram

The figure above shows the relationship among users and use case of the system. It provides an overview of part of the usage requirements for the system and allows description of sequences of events that, taken together, lead to the system doing something useful. Each use case provides one or more scenarios that convey how the system should interact with the users called actors to achieve a specific business goal or function. Based on the diagram above, the author have clearly stated that the earlier project title was called "Web-Based Desktop Environment for C Programming Student", which then changed to "UTP Web Desktop Environment" due to change of the target user of the system during iteration process.

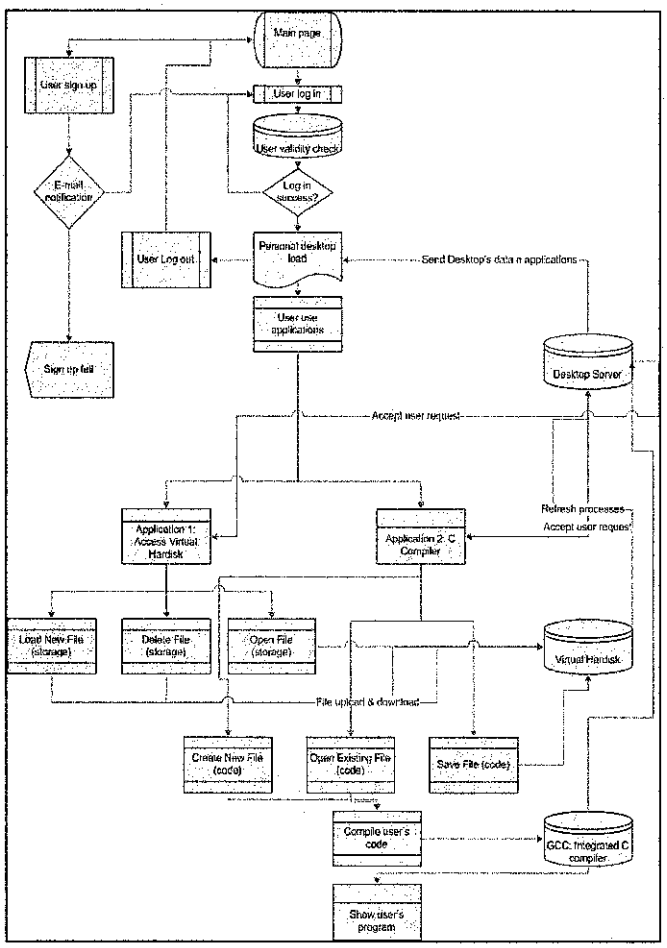

Figure 7: The Data Flow Diagram

The data flow diagram above reveals the relationships among and between the various components in the UTP Web Desktop Environment system. It is an important technique for modeling the system's high-level detail by showing how input data is transformed to output results through a sequence of functional transformations.

3.2.3 Implementation

The implementation phase is one of the time-consuming phases, whereby all the code-generating, programming and system-upgrading will be done. Using the module and also the data flow analysis that were defined previously in the design phase, the generation and development of the system can be started. Because of the nature of the iterative

methodology, the development phase will also be influenced by the changing design phase.

The development also will be modularized, with an early prototype being develop at first. The prototype was developed according to the percentage of the complete system, which means that the prototype can be use, however in a limited functionality. The development will also be depending on the feedback that will be received from the target user during the user review phase. The development and implementation of the system will be discussed in chapter 4.

3.2.4 Test

During this phase, the target audience that had been defined and categorized during the Design phase will be ask to test the prototype that had been develop. The prototype will be made available to the intended user. The users will be expected to perform the usage of the system. The test will be conducted in the user own environments, which can help the system blend in with the environment of each target user.

3.2.5 Deploy

Deploy phase will kick off after all the module had been complete, which means the Deploy phase is the last phase after all the feedback and user review for each prototype had been completed. Deploy phase will not be about prototype anymore, but it will be the phase whereby the real product, in this case, the finished UTP Web Desktop Environment system will be officially completed and launch. There will be no more iterative process after the Deploy phase had been commenced. Nonetheless, there will still be minor updates in order to keep the system stable and reliable with any current situation.

3.1 TOOLS REQUIRED

Hardware

For this project, a Window-based server is required to perform a webhosting to be accessed internally within UTP. The server should be able to handle multiple users simultaneously. In the development phase, a personal computer will be use as a workstation and all the related work will be done using PC before being demonstrated through actual server.

Software

For the software, Asynchronous JavaScript and XML (AJAX) will be used to develop the UTP Web Desktop Environment. AJAX is a programming language that uses both JavaScript and XML to trigger the PHP-coded system.

There are numerous number of system available trough the Internet that using AJAX as their programming language. The author insists on to use EyeOS Microserver Version 0.5.6, an open source system which is released under GPL Version 2, as a pre-installed server that runs PHP 5.2, which is released under the PHP License, version 3.01.

CHAPTER 4

RESULT AND DISCUSSION

4.1 SYSTEM DEVELOPMENT

4.1.1 System Interface Design and Features

This section will describe the current development of UTP Web Desktop Environment with the system interface designs and features provided. Currently, the prototype of UTP Web Desktop Environment version 1.0 is completed with the pre-embedded applications of EyeOS applications. The system is currently running under EyeOS Microserver Version 0.5.6 web server and Macromedia Dreamweaver Version 8 was used in the designing and editing the interface.

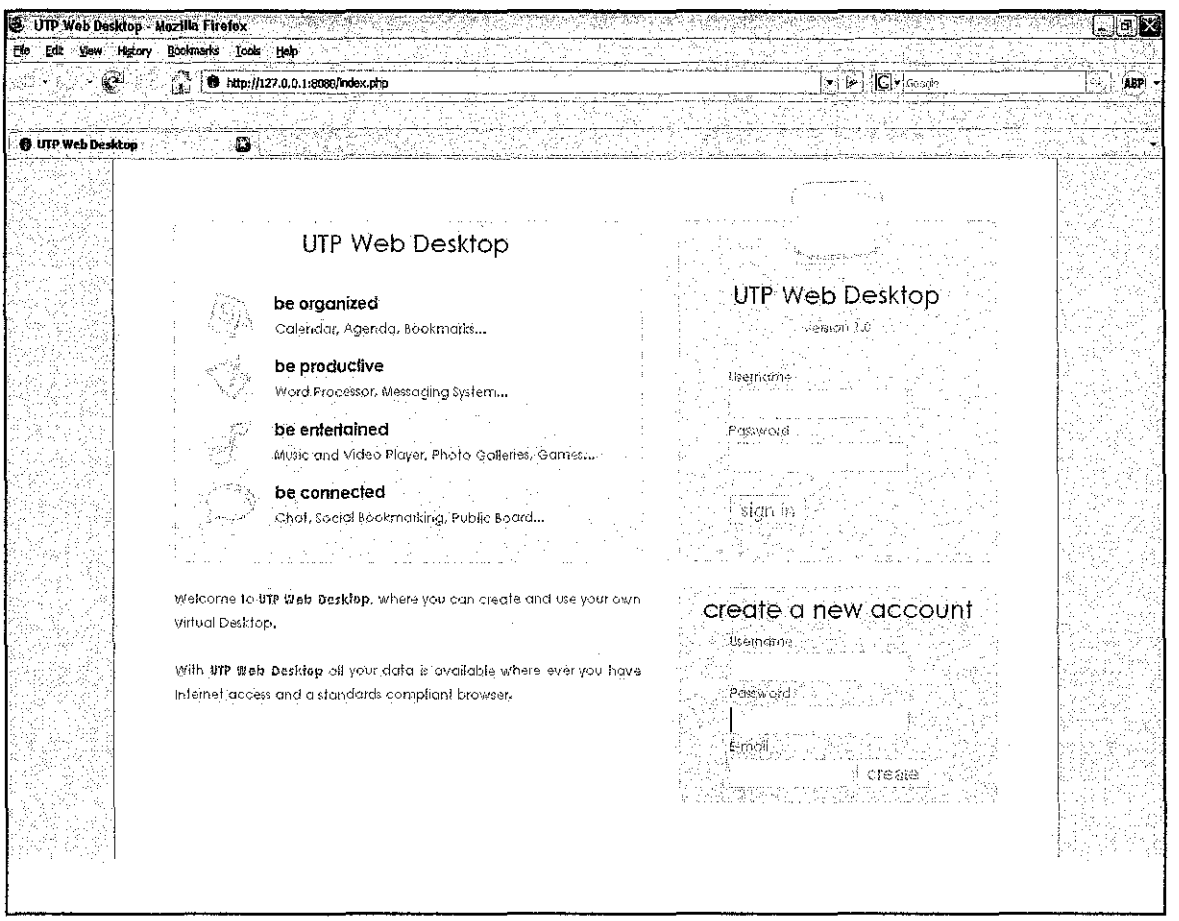

Figure8: UTP Web Desktop Environment Main Page

To put it simply, UTP Web Desktop Environment is similar to the operating system anyone are already familiar with (like Windows, Mac and Linux), except for the fact that it is not stored on the computer that the user are using to access it. In other words, UTP Web Desktop does not need to be installed on user's computer in order to use it. Instead, it lives on a remote system (the web server) that uses the Internet to communicate with the user. Allowing user to access your personal files, safely and securely, from anywhere in the world.

The main page of UTP Web Desktop Environment system (see figure 8) is presented with a user-friendly login box where new user can create a new account by fill up the required fields. Registered user can directly access their workspace by filling up their username and password. Currently, this system can be accessed through UTP

intranet at http://165.0.6.93:8080 using conventional web browsers such as Internet Explorer 5, Mozilla Firefox 2.0, etc.

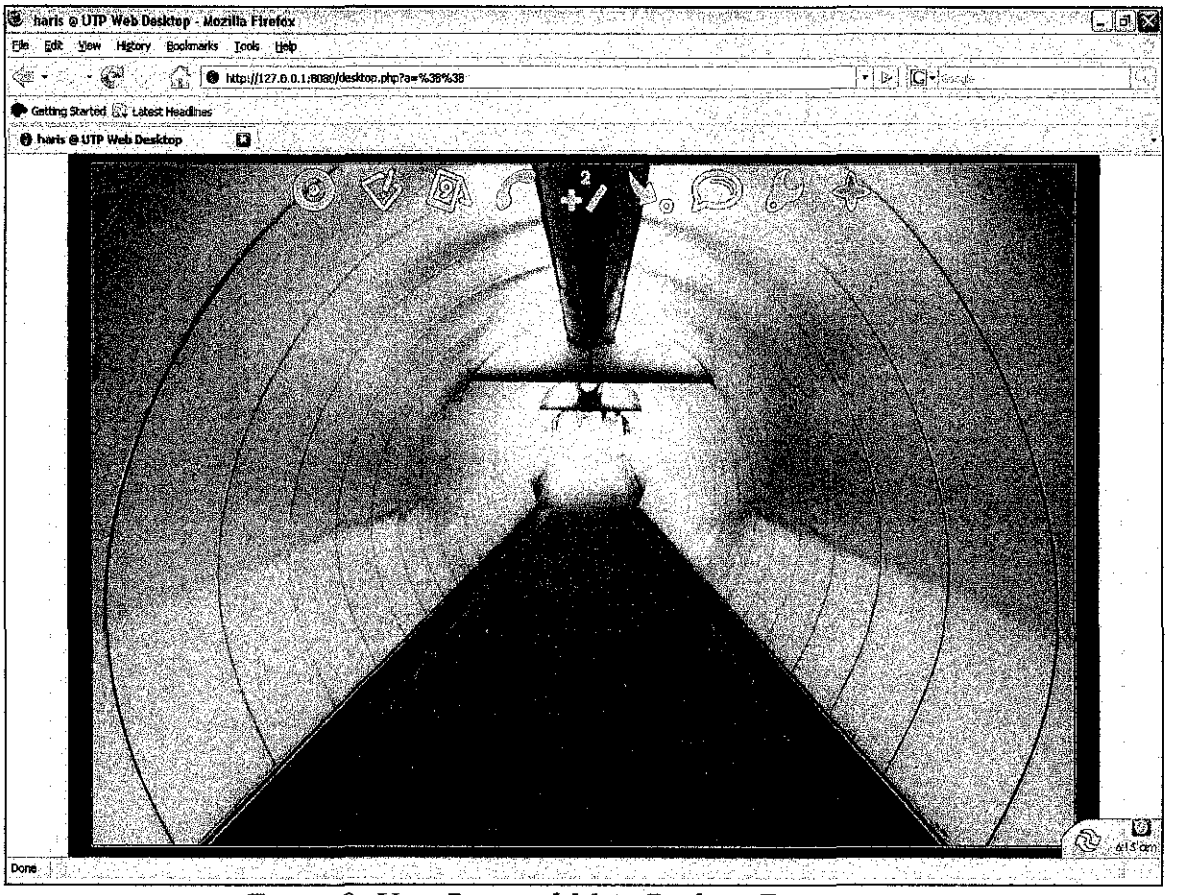

Figure 9: User Personal Main Desktop Environment

Once a user log in to his new account, the user will be presented with the UTP Web Desktop main desktop within his web browser window (refer Figure 9). Just like any other web page, the system allows user to click on icons presented to launch applications and to carry out system function. The desktop's main application toolbar is located along the top margin of the browser window. The system also provided a small group of icons in the bottom-right corner of the window. This contains Recycle folder (also known as recycle bin), system clock, and the Log-out button. The rest of the screen is the user's work area. This is where applications will appear when user opens them from the toolbar. Applications are launched by clicking on their respected icons, located in the main system toolbar. Clicking on these icons will launch one of the many different

tools provided. Just like familiar desktop environment, each application will be presented in its own window within user's web browser.

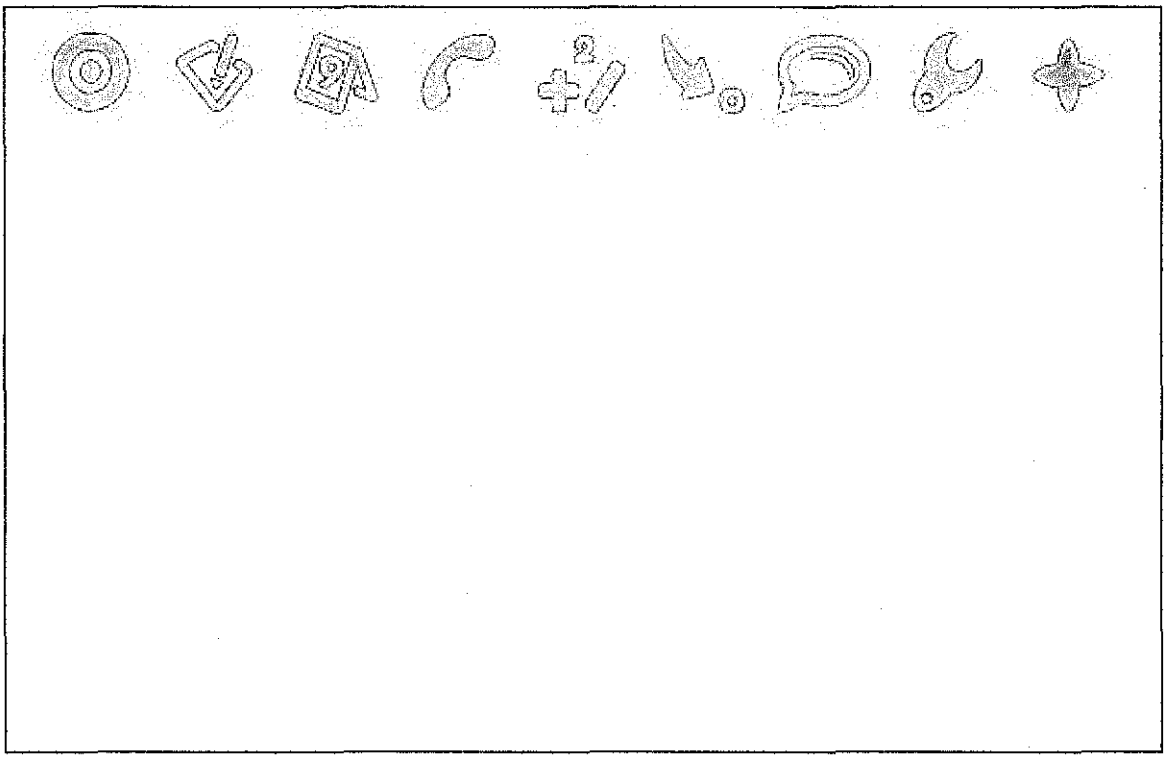

Figure 10: Icons on the main toolbar

Figure 10 shows the icons of applications provided from the system. The first icon is called Home (refer Figure 11). This will acts as user's home directory that contains the files user has created and saved to his UTP Web Desktop account. This system allows user to upload and download any files from the local machines. Within this window, the system also provides user with massage notification where user will be notified if there is any new incoming message.

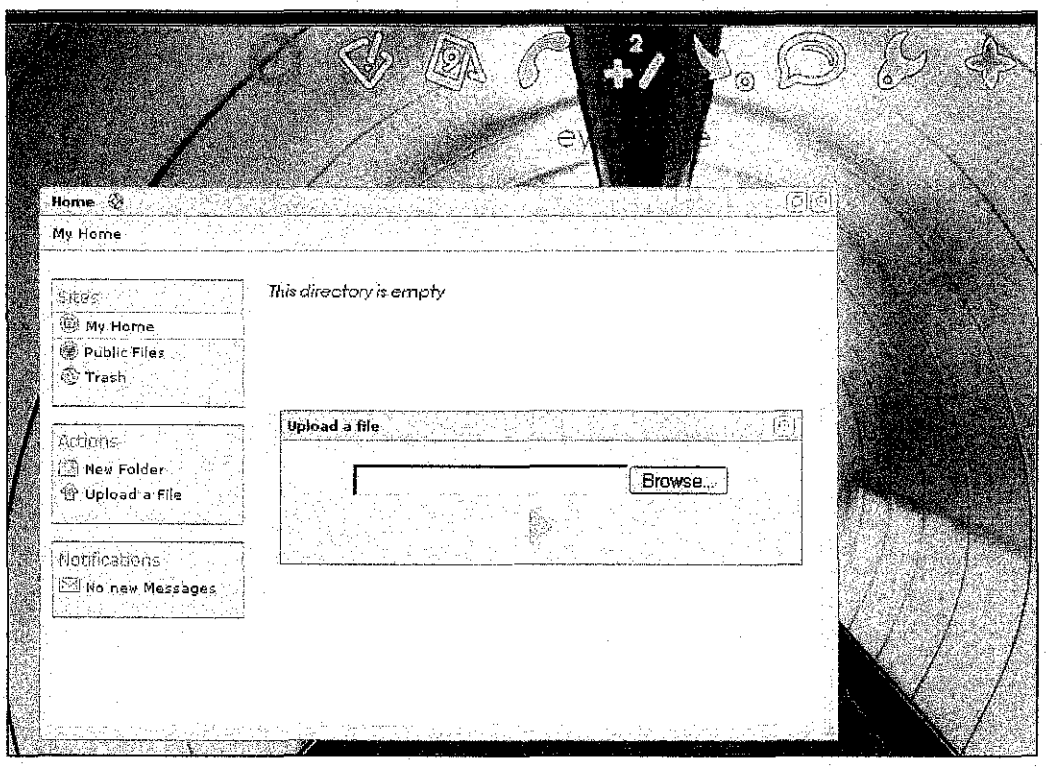

Figure 11: Interface of Home

The second icon is called Word Processor (refer Figure 12) which allows user to create a new file, type all his assignment directly into the web browser, and save it into his account. The file can be saved as .txt or .doc file. Another capability of this word processing is it allows file to be saved as private and public which means the file can be saved as public where other user can access the file. If the file is saved as private, only the author of the file is allowed to view or do changes on it.

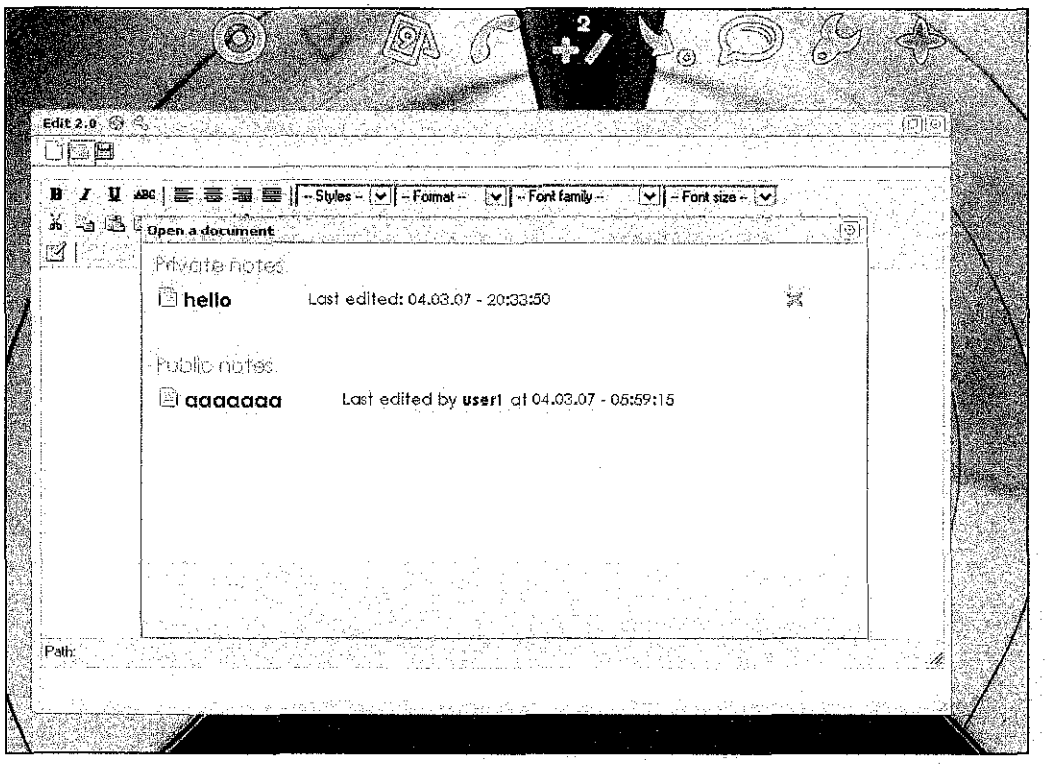

Figure 12: Interface of Word Processor

The third icon will provides user a calendar (refer Figure 13) which user can manage his timetable by registering his to-do-list, appointments, etc directly into the date we wanted.

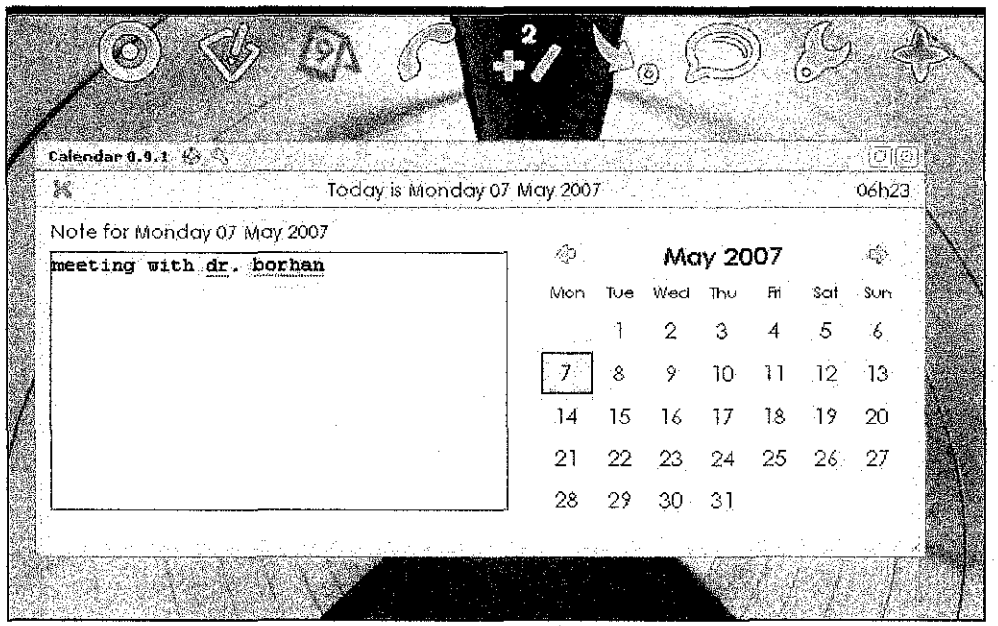

Figure 13: Interface of Calendar

The fifth button is called Calculator (see Figure 14) where user can use its basic functions such as add, subtract, divide, and multiply. The sixth icon will open a new window for user to send a personal message to other UTP Web Desktop Environment users (see Figure 14). The new message notification can be seen on Home window.

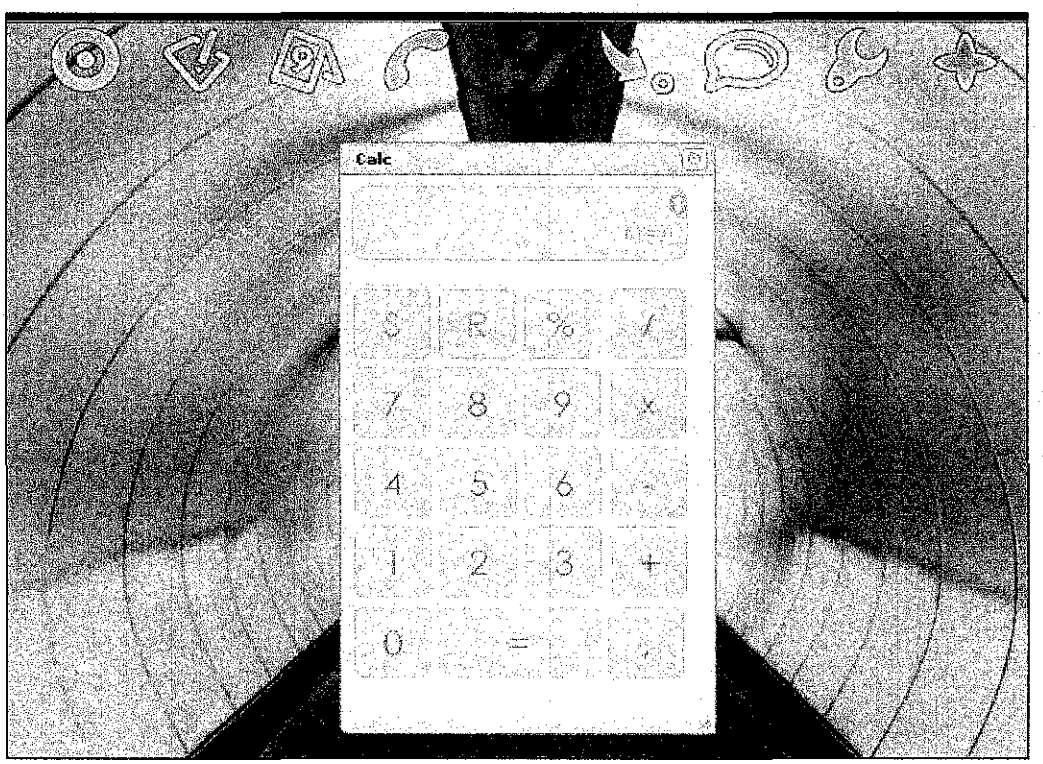

Figure 14: Interface of Calculator

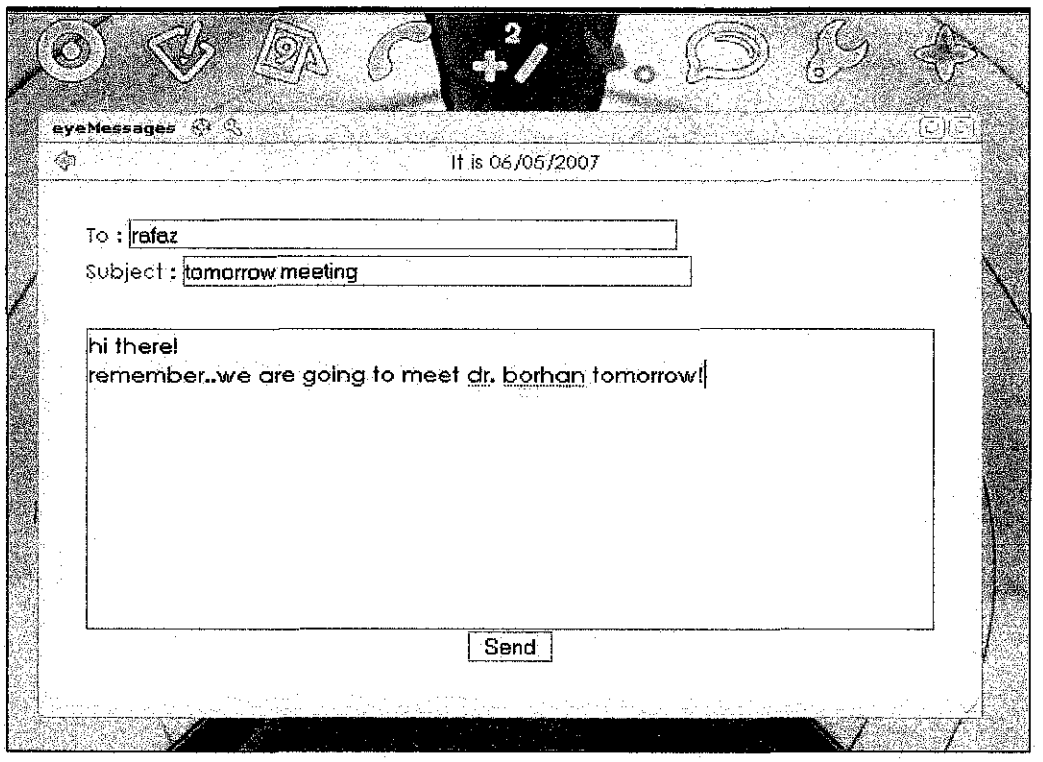

Figure 15: Interface of Personal Message

The next button is called Message Board where user can send any announcements, news, thoughts, etc. that can be seen by all users. This feature is really useful for students to add any announcement such as class replacement, test, or adjunct lectures, etc.

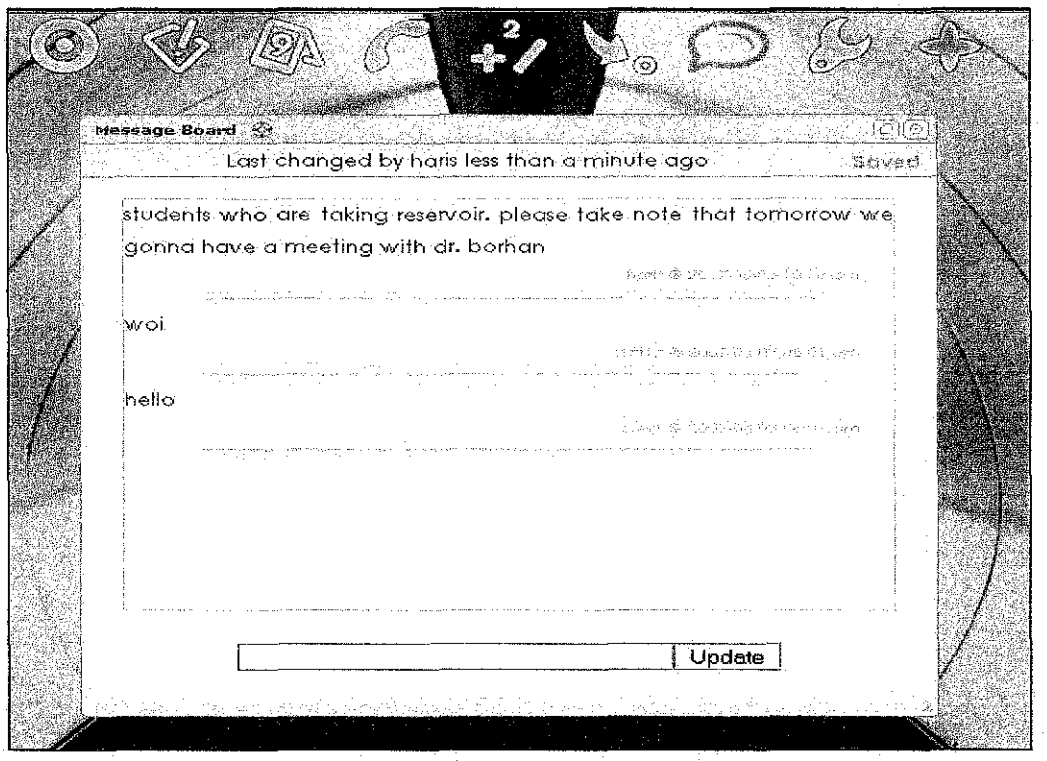

Figure 16: Interface of Message Board

The last two icons are called Options and Applications. The Option button allows user to change the themes of his workspace and password. The Application button is where user can add any new applications created.

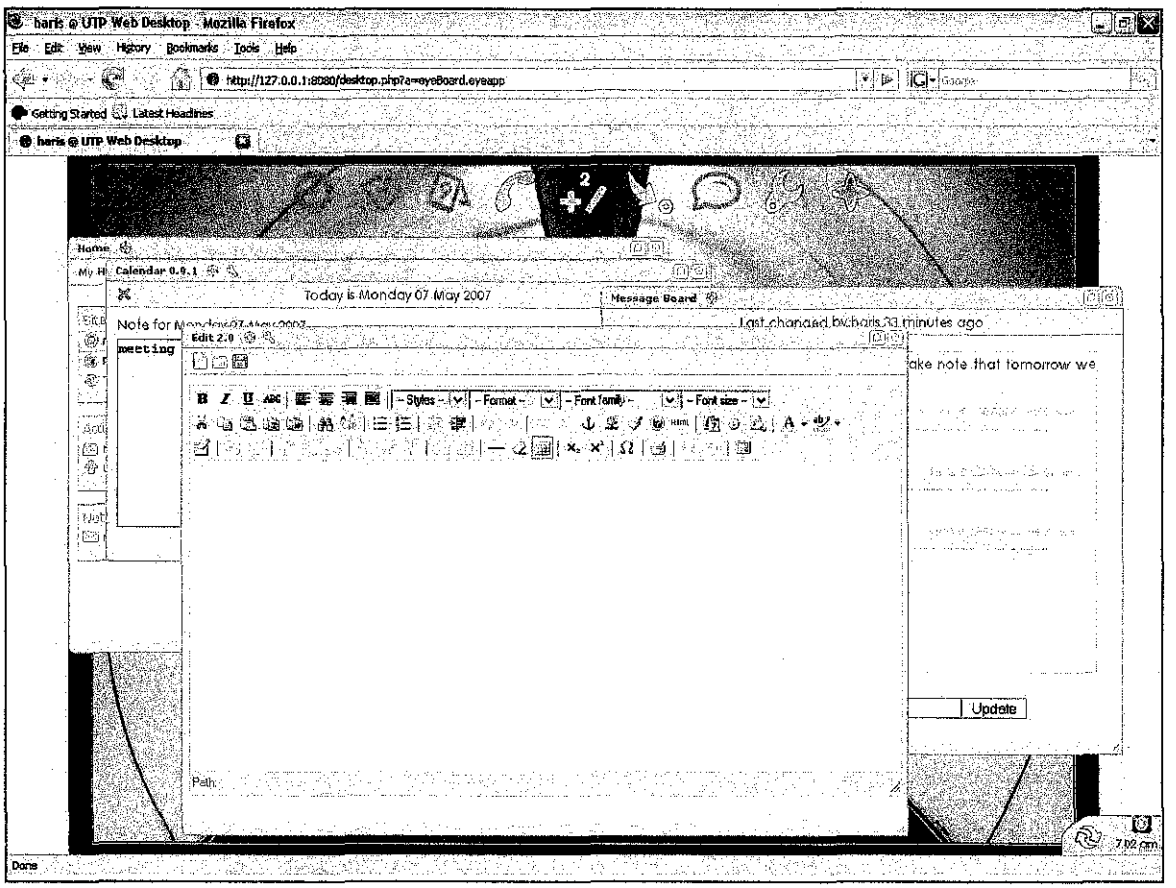

Figure 17: Multiple applications opened at once

Figure 17 shows that the system allows multiple applications can be opened at once. User also allowed dragging and resizing the each window he opened.

4.2 DISCUSSION

Conventional browser-based web applications require the user to submit a request to the server, wait for the server to process the request and generate a response, and then wait for the browser to update the interface with the results. This request-wait-response-wait pattern is extremely disruptive and lowers productivity. Conventional browser-based web applications require the user to submit a request to the server, wait for the server to process the request and generate a response, and then wait for the browser to update the interface with the results. This request-wait-response-wait pattern is extremely disruptive and lowers productivity. High network latency and interface complexity and slow server responsiveness can further impair the user experience, resulting in decrease user satisfaction, and less frequent website visits.

But those weaknesses of conventional browser-based web applications can be eliminated by implementing AJAX style programming because of the architecture of AJAX itself that allows communication between server and client happened in asynchronous way. The figure below expressed on how AJAX working in general.

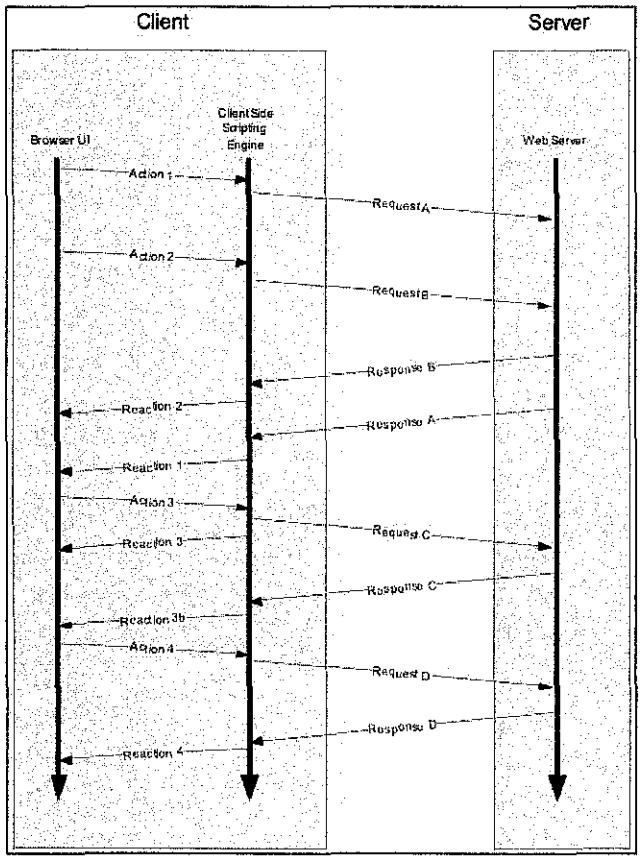

Figure 18: Sample of asynchronous communications using AJAX [16]

This system is developed using AJAX style programming language which are using three core components:

i) JavaScript components.

ii) Data exchange and transformation using the Extensible Markup Language (XML) and Extensible Stylesheets Language Transformations.

iii) A user interface constructed with open standards such as the Dynamic Hypertext Markup Language and cascading stylesheets (CSS).

All the data is stored in file-based system, which means there is no database required to store user's files and action. The system will create a folder for each

registered user and every file the user is accessing, downloading, or uploading is stored in his folder.

4.2.1 JavaScript Components

In order to enable the flexibility of using the system, EyeOS system used four different functions of JavaScript which are called x_core.js, x_drag.js, x_event.js, and x_eyeoswin.js. All of these JavaScript files are simultaneously called whenever user uses the applications. These files react on user request to provide functions for drag, drop, maximize windows, and such.

4.2.2 XML and Extensible Sylesheets Language Transformation

XML files for each application is stored within the PHP files where the files are triggered from user's actions. Each PHP files are contained with the application executable codes. Each XML file is triggered only when user clicked on the respective icons.

4.2.3 xHTML and CSS

All application's setting is stored in CSS files where developer can manage its appearances such as size, location, colors, etc. The system also providing a function that creating CSS file for each user whenever the user makes any change on their desktop environment; which means the system will remember settings (sizes, and locations) of each user. Whenever the user login again, the system will automatically loads user's CSS file and stores back whatever changes the user have made.

4.3 PROBLEMS ENCOUNTERED

For UTP Web Desktop Environment system version 1.0, the author faced many challenges in making the system works as wished.

4.3.1 User Can't Drag and Drop Files onto the Workspace.

For the time being, the author still can't create a function that allows items in the Home directory to be drag and drop into the workspace. There is limitation where the system only allows executable files such as PHP-encoded file that have been modified into eyeOS pre-coded core to be icon in the desktop. But the author positively thinks that the function can be created as future enhancement of the system.

4.3.2 Network Problem

As demonstrated earlier, the system can't be accessed from outside UTP because of the network rules and limitations. Hence the objective of creating this to be mobility as other WWW applications is foiled.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

As the conclusion, we can conclude that the implementation of UTP Web Desktop Environment has provides users with the sense that they were using desktop applications instead of a conventional web-based application. The added value of injecting AJAX into the system might change the others perspective toward the benefits of experiencing an interactive ways of using web applications. The UTP Web Desktop Environment also can be considered as a one step ahead towards a new paradigm of Internet services for UTP student as it combines multiple applications into a single web space. By implementing the project, this will create a new programming environment to all users by allowing them to create their own applications where they can share it with others. It will be a new medium of communications where students can send messages to others in more faster and organized way. After all, the platform of creativity is created, now it depends on student to utilize it.

5.2 FUTURE WORK AND RECOMMENDATION

There are many enhancements that can be made to make UTP Web Desktop Environment more interesting and useful for UTP student.

1) The system should allow UTP student to create his own applications directly from the workspace and share the application with others.

2) User may also be able to create his own themes where he can select the font name, font size, icons, and toolbars.

3) As refer to the problem faced by the author, there should be a future work to make items can be drag and drop on the workspace.

Like the other systems, there must be the updates and modification on the design of the system as we are gearing towards a better world and because user sometimes needs refreshing to make sure they didn't get bored.

REFERENCE

[1] Aaron Weiss (Dec 2005). Introduction to WebOS. *WebOS: Say Goodbye to Desktop Applications,* Retrieved September 2, 2006, from http://portal.acm.org/ft_gateway.cfm?id=1103941&type=pdf

[2] Amin Vahdat, Paul Eastham, Chad Yoshikawa, Eshwar Belani, Thomas Anderson, David Culler (Dec 2005). Architecture of WebOS. *WebOS: WebOS: Operating System Services for Wide Area Applications,* Retrieved September 4, 2006, from http://www.ececs.uc.edu/~yoshikco/papers/webos.pdf

[3] Sean Lockhead, Product Support Group Manager, Kaman Industrial Technologies (1998). Alternatives Technologies of using Internet. *The Use of Internet-Based Technologies – Beyond E-mail and Search Engine,* Retrieved September 5, 2006, from http:// ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=727780

[4] Linda Dailey Paulson (Oct 2005). AJAX Components and Functions. *Building Rich Web Applications with Ajax,* Retrieved September 5, 2006, from http:// www.computer.org/portal/cms_docs_ieeecs/ieeecs/images/ajax.pdf

[5] Keith Smith, Senior Product Manager, Microsoft (May 2006). Introduction to AJAX. *Simplifying Ajax-Style Web Development,* Retrieved September 5, 2006, from http:// ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1631955

[6] Peter G. Kropf (1999). Introduction to WOS. *Overview of the WOS Project,* Retrieved September 7, 2006, from http:// citeseer.ist.psu.edu/508498.html

[7] Slim Ben Lamine, John Plaice and Peter Kropf (2005). Problems of Web Design. *Problems of Computing on the Web,* Retrieved September 9, 2006, from http:// citeseer.ist.psu.edu/benlamine97problems.html

[8] Vittorio Trecordi and Giacomo Verticale (2000). Push and Pull Technology: How it Work. *An Architecture for Effective Push/Pull Web Surfing,* Retrieved September 9, 2006, from http:// eeexplore.ieee.org/iel5/6882/18540/00853679.pdf?arnumber=853679

[9] Denis G. Sureau (Feb 2006). Step by Step of Using RSS. *RSS, Building and Using a Feed* Retrieved September 11, 2006, from http://www.xul.fr/en-xml-rss.html

[10] Denis G. Sureau (Feb 2006). Distributed Content Feed. *Content Feeds with RSS 2.0* Retrieved September 11, 2006, from http:// www.ibm.com/developerworks/xml/library/x-rss20/

[11] Zeppo Network, Inc (2007). Content Management System [ZeppOS – Standard Internet Platform]. New York.

[12] *XIN Widgets.* (n.d.). Retrieved January, 2007, from http://www.xinteleport.com/

[13] *twinklefish WebOS.* (n.d.). Retrieved January, 2007, from http://www.twinklefish.com/

[14] *Framework – AJAX Pattern.* (n.d.). Retrieved January, 2007, from http://ajaxpatterns.org/Ajax_Frameworks

[15] *Ajax Toolkit for PHP – SAJAX.* (n.d.). Retrieved January, 2007, from http://www.modernmethod.com/sajax/

[16] *AJAX Framework.* (n.d.). Retrieved January, 2007, from http://glm-ajax.sourceforge.net/

APPENDICES

APPENDIX 1

FINAL YEAR PROJECT PART 1 SCHEDULE

NO	DETAILS	WEEK NO													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Selection of project topic	■	■												
	-Propose topic		●												
2	Preliminary Research														
	-Project Planning			■	■										
	-Literature Review			■	■	■	■	■							
3	Submission of Preliminary Report								●						
4	Project research/work														
	-Literature Review								■	■	■	■	■		
	-Interface Design											■	■		
5	Submission of Interim Report											●			
6	Oral Presentation												●		

Table 1: Final Year Project Part 1 Schedule

APPENDIX 2

FINAL YEAR PROJECT PART 2 SCHEDULE

NO	DETAILS	WEEK NO													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Programming Research	■	■	■	■	■	■								
2	Web Development														
	- Prototype Development				■	■	■	■							
	- Applications Development								■	■	■	■	■		
	Submission of Progress Report			●											
3	Submission of Final Report Draft							●							
	Submission of Final Report								●						
	Oral Presentation											●			
4	Submission of Project Dissertation												●		

Table 2: Final Year Project Part 2 Schedule