Digital Library of Theses

CHONG FONG FONG
WET 98015

Under the supervision of
Associate Professor Dr. N. Selvanathan

This project is submitted to the
Faculty of Computer Science and Information Technology
University of Malaya
In
Partial fulfillment of the requirement for the degree of
Bachelor of Information Technology

Session 2000/2001

ABSTRACT

Digital Library of Theses is a web-based library system for submitting, storing and disseminating of theses electronically. This system is proposed to overcome the shortcomings of the traditional methods of storing and disseminating of printed theses. Users can use this system to upload their theses in the electronic form to the server and the theses can be retrieved using the efficient search capabilities provided. Source codes can be stored in this system too. This system provides an efficient way of handling source codes by storing the source codes in the reusable components or object form. This is to promote the reusability of the source codes for development of similar projects in the future. A survey has been carried out to explore the issues surrounding the electronic publication of theses and to examine the feasibility of submission, storage and distribution of electronic theses and dissertations. The survey sampled the progress made at other universities in the direction of moving to electronic theses. It also detailed the features offered by other digital libraries. In addition, the justification of the methodology used throughout the development of Digital Library of Theses is explained. The Digital Library of Theses is a web-based application, thus all communications are done via the TCP/IP protocol. All processes are initiated when the user starts the Digital Library of Theses where the connection to the web server as well as the database server is established.

TABLE OF CONTENTS

ABSTRACT i

ACKNOWLEDGEMENT ii

TABLE OF CONTENTS iii

LIST OF FIGURES vii

LIST OF TABLES viii

CHAPTER 1 : INTRODUCTION 1
1.1 PROJECT DESCRIPTION 1
1.2 OBJECTIVES 3
1.3 PROJECT SCOPE 3
1.4 RESEARCH CONDUCTED 5
1.5 PROJECT SCHEDULE 5
1.6 DEVELOPMENT STRATEGY 7
1.7 REPORT OVERVIEW 11

CHAPTER 2 : LITERATURE REVIEW 13
2.1 DEFINITION 13
2.1.1 Theses 13
2.1.2 Electronic Theses 13
2.1.3 Digital Library 14
2.2 TRADITIONAL WAYS OF SUBMITTING, STORING RETRIEVING THESES 14
2.2.1 Problems of Traditional Methods of Submitting, Storing and Retrieving Theses 15
2.3 THE MOVE TO ELECTRONIC THESES AND DISSERTATIONS 16
2.4 EVOLUTION OF ELECTRONIC THESES AND DISSERTATIONS 17
2.5 BENEFITS OF ELECTRONIC THESES AND DISSERTATIONS 22
2.6 ELECTRONIC THESES ISSUES 24
2.6.1 Delivery Format 24
2.6.2 Archiving 27
2.7 FEATURES OF ELECTRONIC THESES AND DISSERTATIONS 30

CHAPTER 3 : SYSTEM ANALYSIS 33
3.1 INTRODUCTION 33
3.2 FACT FINDING TECHNIQUES 33
3.2.1 Interviews 34
3.2.2 Internet Surfing 34
3.2.3 Research 34
3.3 FUNCTIONAL REQUIREMENTS 35
3.3.1 Information Retrieval 35

3.3.1.1 Simple Search 35
3.3.1.2 Advanced Search 36
3.3.2 Display 36
3.3.3 Sorting 36
3.3.4 Theses and Source Code Uploading 36
3.3.5 Viewing and Administering News 37
3.3.6 Sending and Administering Feedback 37
3.3.7 Help 37
3.3.8 Theses Listing 38
3.4 NON-FUNCTIONAL REQUIREMENTS 38
3.5 CONSIDERATION OF PROGRAMMING LANGUAGE TECHNOLOGIES 41
3.5.1 Common Object Model (COM) 42
3.5.1.1 Overview of COM 42
3.5.1.2 COM Servers Types : In Process or Out Process 45
3.5.1.3 Advantages of COM 47
3.5.2 Distributed Common Object Model (DCOM) 48
3.5.2.1 Overview of DCOM 48
3.5.3 Common Object Request Broker Architecture (CORBA) 52
3.5.4 Active Platform 54
3.5.4.1 ActiveX Control 57
3.5.5 Java Platform 57
3.5.5.1 JavaBeans 59
3.5.6 Comparison of ActiveX and Java 60
3.6 CONSIDERATION OF PROGRAMMING LANGUAGES 62
3.6.1 Active Server Page (ASP) 62
3.6.1.1 VBScript 63
3.6.2 Common Gateway Interface (CGI) Applications 64
3.6.3 Comparison of Programming Languages and Components 65
3.7 OTHER CONSIDERATIONS 66
3.7.1 Microsoft Windows NT Server 4.0 66
3.7.2 Internet Information Server 4.0 66
3.8 CONCLUSION 68
3.8.1 Choice of Programming Language 70
3.9 HARDWARE AND SOFTWARE REQUIREMENTS 72
3.9.1 Server Side 72
3.9.2 Client Side 73

CHAPTER 4 : SYSTEM DESIGN 74
4.1 INTRODUCTION 74
4.2 DESIGN TECHNIQUE 75
4.3 USER INTERFACE DESIGN 76
4.3.1 Screen Design 76
4.4 DATABASE DESIGN 80
4.5 SYSTEM FUNCTIONALITY DESIGN 84
4.5.1 Client/Server Architecture 84
4.5.1.1 Three-Tier with an application server 84
4.5.1.2 Digital Library of Theses Three-Tier Client/Server Architecture 85

4.5.2 Data Flow Diagram 86
4.5.2.1 Data Flow Diagram for Theses and Source Code Submission 87
4.5.2.2 Data Flow Diagram for Search Engine 88
4.5.2.3 Data Flow Diagram for Theses Listing 89
4.5.2.4 Data Flow Diagram for Users' Queries and Access 90
4.5.3 System Structure Chart 91

CHAPTER 5 : SYSTEM IMPLEMENTATION 92
5.1 INTRODUCTION 92
5.2 ENVIRONMENT CONFIGURATION AND SETTING 92
5.2.1 Internet Information Server 92
5.2.2 Microsoft SQL Server 7.0 92
5.3 SOFTWARE TOOLS 93
5.3.1 Development Tools 93
5.3.2 Graphical Design Tool 93
5.3.3 Documentation Tools 93
5.4 CODING PRINCIPLES 94
5.4.1 Coding Methodology 95
5.4.1.1 Top-Down Approach 95
5.4.1.2 Bottom-Up Approach 95
5.4.2 Client Side Coding 96
5.4.2.1 Theses Listing 96
5.4.2.2 Theses, Full Source Code or Components Submission 97
5.4.2.3 Search Engine 97
5.5 TESTING 99
5.5.1 Testing Techniques 99
5.5.1.1 White Box Testing 100
5.5.1.2 Black Box Testing 102
5.5.2 Testing Strategies 103
5.5.2.1 Unit Testing 103
5.5.2.2 Integration Testing 105
5.5.2.3 Interface Testing 107
5.5.2.4 System Testing 107

CHAPTER 6 : SYSTEM EVALUATION 109
6.1 PROBLEMS FACED AND SOLUTIONS 109
6.1.1 Lack of Knowledge on Web-Based Programming 109
6.1.2 Lack of Exposure to the Database Server 110
6.2 STRENGTHS OF THE SYSTEM 110
6.2.1 Source Code and Objects Library 110
6.2.2 Electronic Theses Library 111
6.2.3 Year Listing 111
6.2.4 Search Engine 111
6.2.5 On-line Theses, Source Code or Components Submission 111
6.2.6 Downloadable Source Code and Components 112
6.2.7 Attractive and Ease of Use Graphical User Interface 112
6.2.8 Total User Control 112
6.2.9 System Transparency 112

6.2.10 Sorting 113
6.2.11 Validation on Input Data 113
6.3 SYSTEM LIMITATIONS 113
6.3.1 Limited Boolean Combinations For Advanced Search 113
6.3.2 Browser Limitation 113
6.3.3 No Password Encryption 114
6.3.4 No Context Sensitive Help 114
6.4 FUTURE ENHANCEMENTS 114
6.4.1 Enhance Boolean Searches 114
6.4.2 Support Multi Languages Character Set 115
6.4.3 Database Integrity 115
6.4.4 Password Encryption 115
6.4.5 Discussion Forum 115
6.4.6 Provides Interactive Help Function (in demo or video form) 116
6.4.7 Integrating a Mail Server 116

CHAPTER 7 : CONCLUSION 117
BIBLIOGRAPHY 119
APPENDIX 123

LIST OF FIGURES

Figure 1.1	Project Schedule	6
Figure 1.2	Prototype Model	9
Figure 3.1	COM Components in Different Processes	44
Figure 3.2	COM provides the underlying transparent RPC.	46
Figure 3.3	DCOM Architecture	50
Figure 3.4	DCOM's Location Independence	51
Figure 3.5	CORBA ORB Architecture	53
Figure 4.1	Screen Design For Digital Library of Theses	77
Figure 4.2	Digital Library Of Theses Three-Tier Client/Server Architecture	85
Figure 4.3	Data Flow Diagram for Theses and Source Codes Submission	87
Figure 4.4	Data Flow Diagram for Search Engine	88
Figure 4.5	Data Flow Diagram for Theses Listing	89
Figure 4.6	Data Flow Diagram for Users' Queries and Access	90
Figure 4.7	System Structure Chart	91

LIST OF TABLES

Table 2.1	NDLTD Accesses by International Sites	20
Table 2.2	Comparison of Characteristics of HTML, PDF, SGML	25
Table 2.3	Features of Electronic Theses	30
Table 3.1	Comparison of ActiveX and JavaBeans	60
Table 3.2	Comparison Between ASP, CGI and Cold Fusion	65
Table 3.3	Server Side Software Requirements	72
Table 4.1	Client Side Menu	78
Table 4.2	Student Table	81
Table 4.3	Full Source Code Table	82
Table 4.4	Object Table	83
Table 5.1	Software Tools Used For System Development	93
Table 5.2	Software Tool Used For Graphical Design	93
Table 5.3	Software Tools Used For Documentation	93
Table 5.4	Client Side Theses Listing Coding	96
Table 5.5	Client Side Theses, Source Code and Components Submission Coding	97
Table 5.6	Client Side Search Engine Coding	97

CHAPTER 1 INTRODUCTION

1.1 Project Description

Over the years, the undergraduates and postgraduates students are required to produce a printed form of thesis or dissertation and to submit the theses to the faculty as partial fulfillment of their degree requirement. As time goes by, the number of theses submitted to the faculty increased and the maintenance of the theses became a critical problem. The process of submitting, storing and retrieving of the theses involves a lot of manpower, time consuming and wastes storage shelves in the library. Besides, the distribution and reproduction of the theses are limited. A thesis can only be distributed to a person at one time. The theses cannot be distributed to a large number of target markets concurrently.

However, with the recent development of the web technology, a brand new area on information distribution in a more effective way is introduced. Therefore, the Digital Library of Theses is developed to go beyond the capabilities of the printed form of theses. It provides a better way of submitting, storing and retrieving of theses. It also provides an effective management, administrative and dissemination of the theses in a university. The Digital Library of Theses also serves as one of the convenient ways of obtaining information about the research works done in the university.

The Digital Library of Theses serves to enable the users to access the theses' database anytime and thus enhancing the availability of the information. This feature is exceptionally true when a user wishes to search for a particular thesis in the library.

With this system, the users need not have to go to the document room to search for a thesis. This will definitely relieves the user from having to constantly be at the document room to search and read the desired thesis. All this can be done through the Digital Library of Theses via the Intranet. Users can read, print and download any chapters of the thesis, which is of interest to them.

In addition, the Digital Library of Theses only accepts theses in the electronic form. Students need to upload their theses in the electronic form to the server. A search engine is provided to facilitate users in retrieving the theses stored in the server. In order to solve the issue of archiving, theses are written to CD-ROMs periodically for long-term preservation. Therefore, the search engine can perform a search through the database as well as through the CD-ROMs.

Another important feature provided by the Digital Library of Theses is storing and retrieving source codes. Traditionally, after developing a system, students will submit their source codes in diskettes or CDs without proper description. In order to handle and manage the source codes more efficiently, source codes are now broken into reusable components or object with proper description when they are uploaded to the server. This also promotes the reusability of the source codes in developing similar projects in the future. With the search capabilities provided by this system, users can retrieve the source codes for future use.

One of the most interesting features of the Digital Library of Theses is an attractive and user-friendly interface that can accommodates multi-level of users ranging from the novice to the professional.

1.2 Objectives

The objectives of the Digital Library of Theses are as follow:

- To increase the availability of student research for scholars and to preserve it electronically
- To lower the cost of submitting and handling theses
- To improve graduate education by allowing students to produce electronic documents, use digital libraries and understand issues in publishing
- To empower universities to unlock their information resources
- To improve access to the information in graduate research by making it immediately available to a large audience through the Internet
- To save the storage space in the library
- To provide an attractive and easy-to-use graphical user interface that accommodates all level of users
- To extend the roles of traditional theses to on-line search, on-line help and on-line submission
- To enable users to have unrestricted access to abstracts and contents anytime

1.3 Project Scope

The Digital Library of Theses is developed to serve within the Faculty of Computer Science and Information Technology. Therefore, the users of this system consist of all the students in the Faculty of Computer Science and Information Technology. The users can have access to functions, which includes on-line search engine, theses

listing, announcement, feedback and on-line help. The users will be granted with unrestricted access to the abstract and the full text of the electronic theses stored in the digital library. The theses are delivered in PDF format. Only those users who are going to submit their theses to the faculty are allowed to access to the on-line theses submission function.

The Digital Library of Theses is developed to provide the following functions:

- The core function of the Digital Library of Theses is to serve as a web-based electronic theses library system. Thus, it will cover various types of search criteria and provide different kinds of search options. Search results are displayed in both the theses listing and full text display view.

- Other functions include electronic theses and source codes submission. In future, users are required to upload their theses and the source codes to the server. The traditional way of theses and source codes submission will no longer being used.

- This system provides function that enables a user to select a particular thesis from the year indexing, author indexing and department indexing. The theses will be displayed according to year, author's name and department name.

- This system provides other functions like on-line help, feedback and announcement. Users can view an announcement posted by the administrators or send their comments to the administrators. They can also use the help function to assist them in using the system.

1.4 Research Conducted

Throughout the development of the Digital Library of Theses, research in several areas has been conducted. These include:

- On-line library information retrieval system

  This is to find out the requirements of an on-line information retrieval information.

- Web server and database server

  This is to understand how to administer and to communicate with a web server and a database server.

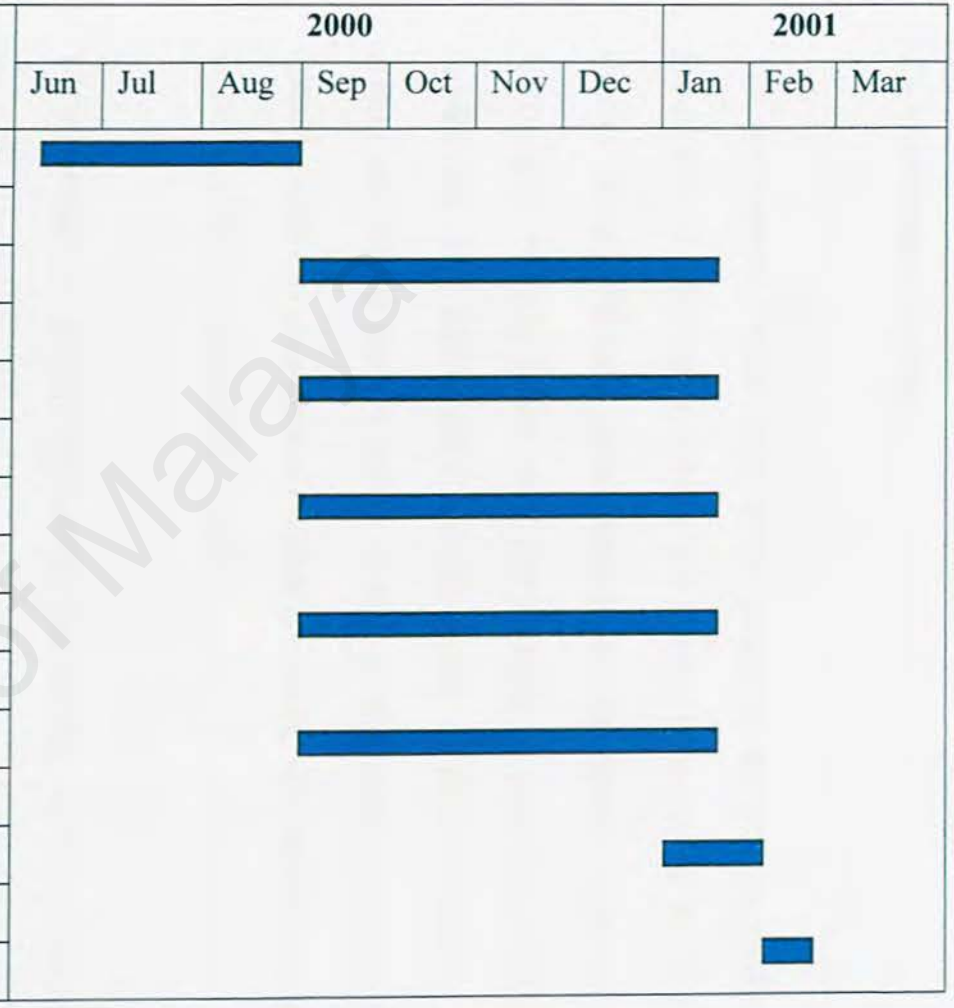
- Reusable components

  This is to understand how to store source codes in the library in the form of reusable components or objects. The reusability of the components stored is also in the research areas.

1.5 Project Schedule

Figure 1.1 illustrates the schedule for the development of the Digital Library of Theses.

No	Phase	Duration	Start	Finish
1	Requirement Analysis	75 days	10.6.2000	31.8.2000
2	Quick Design	120 days	1.9.2000	15.1.2001
3	Construction of Prototype	130 days	1.9.2000	15.1.2001
4	Prototype Evaluation	130 days	1.9.2000	15.1.2001
5	Refine Requirement	130 days	1.9.2000	15.1.2001
6	Engineer Product	130 days	1.9.2000	15.1.2001
7	Documentation	25 days	30.12.2000	23.1.2001
8	Presentation Preparation	10 days	1.2.2001	10.2.2001



Task

Figure 1.1 Project Schedule

1.6 Development Strategy

The development strategy used in this project is the software prototyping model. Prototyping of information systems is a worthwhile technique for quickly gathering specific information about users' information requirements. A prototype information system is a working model of a system built to learn about the system's true requirement. This development strategy quickly develops a working model of the system and uses the model to enable users to get the feeling of how the system should operate. It is not a complete system, since it is being built quickly; only some essential functions will be included in the model.

The advantages are given consideration when deciding whether to prototype. Some of the major advantages of the prototyping that support the reason why the development model is used are:

- The system being developed should be a better fit with users' needs and expectations. Building a system by using prototype is an interactive process whereby continuous evaluation and revision of the model system is carried out until the users and developers fully understand the requirements of the system. The prototype can be used as an interactive tool that shapes the final system to accurately reflect users' requirements and ensure that the final product will address the users' needs and expectations more closely.

- The prototype can be changed many times. As with any systems effort, early changes are less expensive than changes made late in the project's development.

Although the prototype represents an investment of time and money, it is always considerably less expensive than a completed system. This will reduce the cost of the system.

- The prototype create a more accurate idea of what the users really need as the user get to see a tangible model of the system at an early stage of development rather than trying to visualize the abstract specification. System problems and oversights are much easier to trace and detect in a prototype with limited features and limited interfaces than they are in a complex system. This helps the user to provide useful feedback and to correct misconception and errors on the project before too much effort is wasted.

- Building prototypes is not a difficult task with various kinds of DBMSes and 4GL programming language in the market. All these tools permit speed of construction that is impossible with traditional programming.

- Prototyping is inexpensive as compared to the conventional model.

Prototyping consists of six steps as shown in Figure 1.2

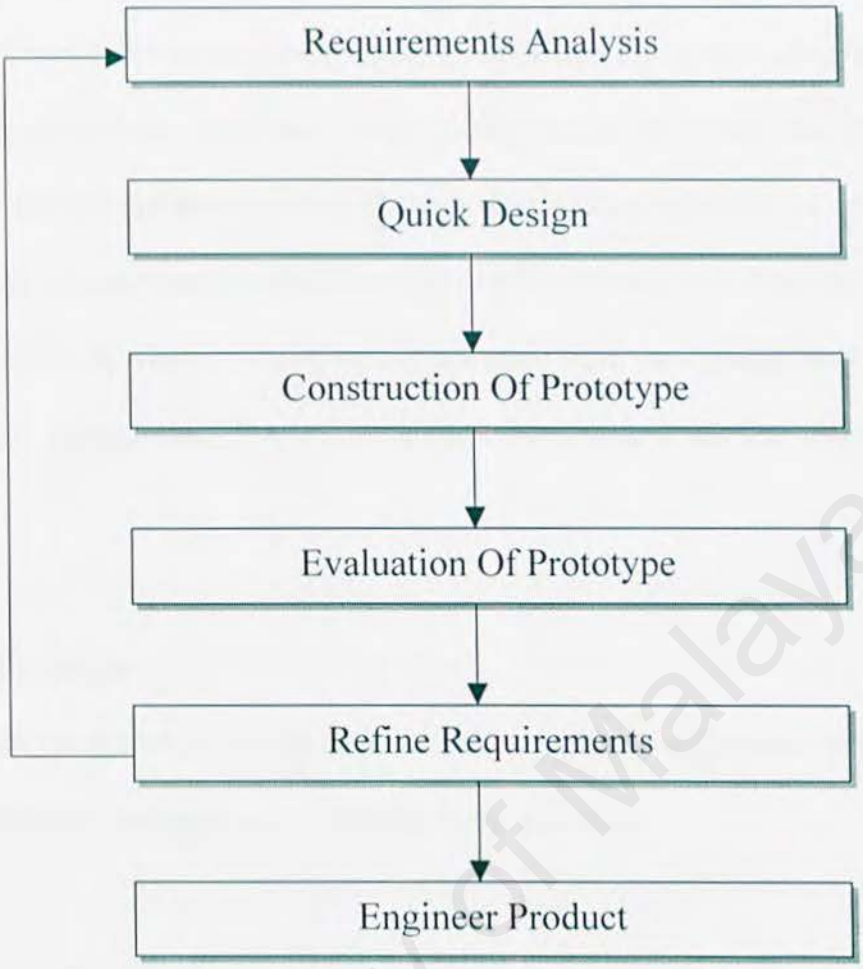


Figure 1.2 Prototype Model

Requirements Analysis

The prototype phase begins with requirements gathering. This is the phase where developer and users meet to define the overall objectives for the system, identify whatever requirements are known and outline areas where further definition is crucial.

Quick Design

Quick design focuses on the representation of those aspects of the system that will be visible to the user such as input approaches and output formats. There are many design techniques that can be used.

Construction Of Prototype

Based on the representation on the system in the quick design phase, the prototype is then being constructed. This is the part where coding occurs. However, this phase does not include the full functional coding of the system. Codes written in this phase are non-functional meaning that it only allows the user to feel the flow of the system. All data and information shown to users are not retrieved from the database and they are hard coded in the program. The actual coding of the system is at the engineer product phase.

Evaluation Of Prototype

The users evaluate the prototypes being constructed for necessary refinements needed. The users detect system problems and oversights in the prototype.

Refine Requirements

All necessary changes in the requirements after evaluation of prototype are done in this phase. Changes in the prototype should move the system closer to what users say is important. Each modification necessitates another evaluation. Iteration occurs as the prototype is tuned to address the users' needs and the users' requirements are thoroughly understood by the developer.

Engineer Product

All requirements are correctly defined and agreed upon by both the developer and the users at this stage. The actual coding of the system is at this stage. The work that is performed here is to finish with the product engineering.

There are various types of prototyping such as the experiment prototype, throwaway prototype, evolutionary prototype and user interface prototype. The prototype model used in the development of this project is the evolutionary prototype. An evolutionary prototype is designed to be adapted for permanent use after the ideas are clarified and must be built using the programming tools that will be used for the final system.

1.7 Report Overview

The purpose of this report is to document all the research findings and phases of developing the Digital Library of Theses. The report is divided into four chapters.

Chapter 1 Introduction

This chapter covers the description of the project, including its objectives, scope, schedule and development strategy being used in developing the system.

Chapter 2 Literature review

This chapter discusses on the research done on the topics, which are relevant to the project. It includes definition of the electronic theses, evolution of the electronic theses, evolution of the traditional ways of storing and retrieving theses to the new web-based approach.

Chapter 3 System analysis

All the requirements of the system including both the functional and non-functional as well as the consideration of the development technologies are discussed in this

chapter. It also includes the reasons why certain platform and development technologies are chosen.

Chapter 4 System design

This chapter focuses on the design of the system including the system functionality design and user interface design. Data flow diagrams and a system structure chart are presented.

CHAPTER 2 LITERATURE REVIEW

2.1 Definition

2.1.1 Theses

A thesis is a lengthy research work written to prove a theory or proposition. There are several different types of thesis. Theses include substantial work in often tightly defined fields of study, and therefore they often contain material unavailable elsewhere. Often they are never published, so reading the original thesis may be the only way to obtain certain information. Theses can be written to obtain a Masters or Doctoral degree. These are known as a Major Thesis. Some (generally shorter ones) are written as part of a degree, and are known as a Minor Thesis. Others are also written as part of the requirements for obtaining an honors degree. Regardless of the length or name, any thesis must display a high level of scholarship and research to be accepted by a university.

2.1.2 Electronic Theses

Electronic theses and dissertations are defined as those theses and dissertations submitted, archived or accessed primarily in electronic formats. That includes traditional word-processed (or typewritten and scanned) documents made available in Print Document Format (PDF), as well as less-traditional hypertext and multimedia formats published electronically on CD-ROM or on the World Wide Web.

2.1.3 Digital Library

There are many definitions of a "digital library." Terms such as "electronic library" and "virtual library" are often used synonymously. The elements that have been identified as common to these definitions[1] are:

- The digital library is not a single entity;
- The digital library requires technology to link the resources of many;
- The linkages between the many digital libraries and information services are transparent to the end users;
- Universal access to digital libraries and information services is a goal;
- Digital library collections are not limited to document surrogates: they extend to digital artifacts that cannot be represented or distributed in printed formats.

A digital library is a collection of information that is stored and accessed electronically. The purpose of a digital library is to provide a central location for accessing information on a particular topic. A digital library should also have a user interface that is easy to use.

2.2 Traditional Ways Of Submitting, Storing and Retrieving Theses

For most scholars, the graduate thesis or dissertation is the first major work of scholarship they produce. Most of the theses are prepared on word processors. The theses submitted are in printed form. Most universities world-wide require print copies of the theses to be deposited in the Library for circulation and archival. Many libraries use Interlibrary Loan to respond to requests for access to these theses or

dissertations. Currently, in order to access the full text of a print thesis or dissertation, researchers need to procure it from the library of the university where it was produced, either in person or through interlibrary loan. Some schools, however, do not participate in interlibrary loan, forcing some researchers to travel great distances to access those scholarly works. Where dissertations and theses are archived by UMI, researchers can buy them in print, microfiche, or microfilm formats for a fee.

Faculty of Computer Science and Information Technology, like most universities world-wide requires students to produce theses in partial fulfillment of their degree requirement. The theses are submitted in printed form and the source codes are submitted in diskettes or CDs. The theses and the diskettes or CDs are then stored in the document room. Students who need to access the full text of a printed thesis have to go to the document room. Over the years, as more theses have been produced, the traditional methods of storing and retrieving theses become a serious problem.

2.2.1 Problems Of Traditional Methods Of Submitting, Storing And Retrieving Theses

Traditionally, theses and dissertations have consisted of written texts produced and bound in book or document form and archived and available through the library. Traditional methods of archiving and storing theses and dissertations are inefficient and unwieldy. Good management of theses and source codes are undermined Many theses and dissertations lie moldering in library basements, with no efficient way for researchers or students to locate the information that may be contained in them. The

storing of theses in printed form can also be a waste of storage space in the document room as more and more theses are stored over the years.

In addition, the diskettes and the CDs which contain source codes are not well managed and without proper names and descriptions on the labels on the diskettes or CDs. Thus, the source codes cannot be re-used in development of other similar projects. It is useless to have resources that cannot be used in the future.

Furthermore, time involved in searching a thesis is long. Students have to go to the document room and spend a lot of time to search for a particular thesis due to lack of efficient search capabilities provided. Besides, most of the scholars and students find the required and allowed text-based thesis or dissertation insufficient to fully document and represent their work. Moreover, the cost involved in producing print copies for the theses is high

2.3 The Move To Electronic Theses and Dissertations

With the emergence of the Internet as an international public utility that is sustainable, electronic theses became viable. The World Wide Web is an easy-to-use Internet service to popularize this form of computer-based networked communication for a mass audience.

The impact of the growing number of users and services in the Internet has encouraged universities to restructure their way of handling theses as well as providing better services to users both in terms of information access, such as

searching for theses and in administration work which include storing and archiving. The Internet Phase from 1990 to 1995 is viewed as an electronic theses development because the Internet is an important component of the infrastructure that supports electronic theses.

Many universities and libraries have taken an interest in moving toward publication of graduate students' work on World Wide Web or individual scholars might publish their own works on the Web, thus allowing free access to full texts. Universities have a prime responsibility to record and archive their graduate students' works. Many universities are now in the process of digitizing information in an effort to preserve it and to make it more widely available. Many are now making digitised versions of traditional (print) theses available on-line. In some universities it is now mandatory to submit and publish academic work electronically; some even go so far as to completely eliminate printed copies

The motivations for this change from traditional printed copies to electronic formats include greater access for the academic community, more creative freedom through various forms of hypermedia, and the potential for more interactive, collaborative work.

2.4 Evolution of Electronic Theses and Dissertations

The concept of electronic theses and dissertations (ETDs) was first openly discussed at a 1987 meeting in Ann Arbor arranged by UMI, and attended by representatives of

Virginia Tech (Ed Fox from Computer Science and Susan Bright from the Computing Center), University of Michigan, SoftQuad, and ArborText.

In 1992, the Coalition for Networked Information sponsored a project discovery workshop with 11 invited universities, each of which had documented the interest of their graduate school, library and computing/information technology groups. This meeting was planned by representatives of UMI, Council of Graduate Schools, and Virginia Tech. Subsequently, a number of further discussions were held at CNI meetings. In connection with one of these, representatives from UMI and Virginia Tech visited Adobe, to learn about plans for the Adobe Acrobat family of tools.

Late in 1995, Virginia Tech prepared a pre-proposal to the U.S. Department of Education regarding a three year effort to build the NDLTD, and also requested that SURA fund initial work on establishing a part of the Monticello Electronic Library for ETDs for the Southeast. The first of these led to funding September 1, 1996 and the latter covered calendar year 1996 pilot efforts in the Southeast. North Carolina State University was the first institution seeking to join the initiative, and initial electronic submissions are expected there in October. The first regional workshop for Southeastern universities was held August 1-2, 1996, hosted by University of North Carolina, Charlotte. Many discussions have been held, and presentations given, in the region, nation, and even internationally. There appears to be interest in such institutions as: Auburn, Clemson, Georgia Tech, Michigan State, Mississippi State, MIT, Oklahoma State, University of Georgia, University of Utah, University of Virginia, and Vanderbilt.

Since then, the Networked Digital Library of Theses and Dissertations (NDLTD) has been under construction. The Networked Digital Library of Theses and Dissertations (NDLTD), funded by a grant from the U.S. Department of Education, is a collection focused specifically on digitized versions of theses, dissertations, and technical papers that began in 1996 at Virginia Tech. The NDLTD reports that more than twenty universities around the world have become official contributing members of the Initiative in 1998, and nearly twice that number have expressed interest or are taking steps to participate. NDLTD makes theses and dissertations available free on the Web, and many libraries and universities offer computer access to the World Wide Web.

The University of Virginia has begun testing and adapting a distributed storage and retrieval system developed by Jim Davis of Xerox and Carl Lagoze of Cornell University that will allow researchers to use the Web to browse the entire Networked Digital Library of Theses and Dissertations by author, subject, keyword, department, or year of publication. That system will also allow users to search and retrieve chapters or sections of a thesis or dissertation to home in on specific sections that are of interest to them.

Table 2.1 shows the NDLTD accesses by International Sites. There is an increase in access in most of the countries. This proves that other countries in the world are interested in this initiative too.

[Sorted by 1999-00 percentage of increased accesses.]

International Domain	1997	1997 rank	1998	1998 rank	1997-98 Increase	1999	1999 rank	1998-99 Increase
Mexico	102	46	438	34	329%	6,401	7	1,361%
Malaysia	439	13	1,432	16	226%	12,022	3	740%
Taiwan	311	24	1,091	20	251%	7,694	5	605%
Japan	495	16	2,456	12	396%	11,881	4	384%
Spain	176	27	1,844	14	948%	6,344	8	244%
Canada	2,367	4	3,970	5	68%	12,975	2	227%
Indonesia	113	32	1,826	13	1516%	5,642	11	209%
Singapore	653	13	1,732	11	165%	5,270	12	204%
Thailand	967	8	1,089	19	13%	3,129	16	187%
Finland	432	18	1,098	21	154%	3,081	17	181%
Brazil	1,130	7	1,449	7	28%	3,939	14	172%
United Kingdom	2,922	1	8,170	1	180%	16,955	1	108%
Italy	725	12	2,553	10	252%	5,183	13	103%
Greece	958	9	1,414	15	48%	2,117	23	50%
Australia	2,501	2	4,223	4	69%	6,167	9	46%
France	1,161	6	4,431	3	282%	5,822	10	31%
South Korea	1,264	5	2,201	8	74%	2,264	22	3%
Netherlands	867	10	2,781	9	221%	2,646	19	-5%
Germany	2,378	3	7,373	2	210%	7,205	6	-2%

Table 2.1 NDLTD accesses by International Sites

Companies such as University Microfilms International (UMI), IBM, and Online Computer Library Center (OCLC) are participating in the unfolding of the NDLTD. Thus, IBM donated a large SMP computer that will serve as the central host for this effort, and which can run IBM digital library software. Various IBM products for handling databases, image collections, searching on image content, and rights management have great potential for helping with the NDLTD.

UMI has a representative on the NDLTD's steering and technology committees and has cooperated with Tech since the start of the project, continuing to be active with the NDLTD while developing its own service. In addition, UMI is committed to using and promoting the applications and standards developed by the NDLTD.

There have been initiatives by various project groups besides Virginia Tech, notably the European Initiative in Library and Information Aerospace - EURILIA [2], the world renowned UMI (formerly University Microfilm International) [3], and Virginia Polytechnic Institute amongst others. The projects have a multiplicity of aims and objectives including enhancing teaching, research, and information dissemination. Cost reduction to students, researchers and libraries is also mentioned as an advantage of electronic theses.

The Library of Congress's National Digital Library Project plans to digitize five million items by 2000, and many university, public, and private libraries worldwide are currently working on digitizing their collections as well. EURILIA started in February 1994, and it attempts "to establish a pan-European system for information access, retrieval, image browsing and document delivery". EURILIA's initial focus is on the provision of access to academic theses.

A University Thesis On-line Group (UTOG) also exists in the United Kingdom. UTOG is investigating the technical, cultural and administrative issues involved in the storage and delivery of theses in digital form. Recent discussions in the DIGLIB Listserv revealed that institutions in Holland, Canada, and Australia are also involved, or are interested, in electronic theses and dissertations.

True success in these projects will potentially mean a permanent change in graduate education and scholarly publishing, with digital libraries playing a more dominant role in supporting and disseminating research.

2.5 Benefits of Electronic Theses and Dissertations

- **Students**

The electronic theses project is primarily an effort to improve graduate education so that graduate students become information literate, learning how to become electronic publishers and knowing how to use digital libraries in their research. With access to the electronic theses system, graduate students will be able to find the full texts of related works easily, to read literature reviews prepared by their peers, and to follow hypertext links to relevant data and findings. Their professors will be able to point to the best examples of research in their area, even to the level of an interesting table, an illustrative figure, or an enlightening visualization. Also, students can benefit by learning how to become electronic publishers, preparing them for their future work. Since this educational initiative targets all graduate students, it is unique in its potential to train future generations of scholars, researchers, and professors. If they can publish electronically and add to digital libraries, future works they write will not have to be scanned or re-keyed. Finally, students are likely to benefit financially from the electronic theses. Publishing electronically should save them the costs of preparing at least some of the paper copies now required. There also may be lower fees from their university and other parties for filing their final copy.

- **Universities**

Few universities have a university press, and many of those are not profitable. Yet, through the electronic theses library, every university can publish the results of its graduate students with a minimal investment. This should increase university prestige, and interest outsiders in the research work undertaken. University libraries can save shelf space that would otherwise be taken up by theses, and the costly handling of paper theses by personnel in the graduate school and library can be reduced or eliminated.

- **Research and Publishing**

Student research should be aided by the electronic theses library since graduate students will have a single repository for the work of their peers, supported by full-text search. Other researchers, including people in companies interested in opportunities for technology transfer, can look to the electronic theses library as a way to quickly learn of new findings. Through the electronic theses library, universities can control one important class of the intellectual property they produce, and can share it freely with other universities to reduce overall costs. This is of particular interest to universities, which now cannot control what happens to the research publications they support, and later spend large sums to buy back research publications from commercial publishers.

- **Archiving**

Archiving theses and dissertations electronically can help to alleviate some of the problems involved in storage, and making full-text versions available either on the Web or as e-mail attachments would make access almost immediate. It is more

efficient to allow theses and dissertations to remain in digital formats to facilitate ease of access and handling Electronic versions on disk, CD-ROM, or other digital electronic media could be cheaper as well.

2.6 Electronic Theses Issues

2.6.1 Delivery Format

Delivery Format is an important issue for electronic theses. To satisfied readers, one of the factors is the appropriate delivery format is used. Different electronic theses use different types of formatting for the best presentation.

- **PDF**

PDF (Portable Document Format) is a proprietary format developed by Adobe for the purpose of electronically published documents to appear as created on multiple platforms. PDF compresses the file and the size of file becomes smaller. It is widely used for transmission and distribution of documents over networks. PDF supports hypertext linking. PDF grow out of PostScript and was developed specifically for World Wide Web presentation. A PDF file can be easily created in WordPerfect or MSWord by printing the file to a PDF Writer. Once turned into a PDF, a file can only be read using an Adobe Reader. Alternatively, a user can save their file as PostScript and use the Adobe distiller to create a PDF file. PDF is a widely used format in the field of electronic publishing.

- **TeX and LaTeX**

TeX and LaTeX were developed to present complex mathematical formulas and, consequently, are good tools to use in developing electronic versions of technical documents that rely heavily on math equations and figures. TeX and LaTeX are supported on multiple architectures and the software is free.

- **SGML**

SGML (Standard Generalized Markup Language) is an international standard (ISO 8879) developed to facilitate the interchange of electronic text. Using SGML, mark-up languages can be developed that can control the structure and content of electronic documents. SGML has great potential because it is not dependent on any platform, application or format.

To use SGML a DTD (Document Type Definition) is developed. A DTD is a set of tags that defines the content and relationships within a given document. The DTD works as a grammar, or set of rules, that can be understood by any application that accepts SGML. A DTD can easily be checked for errors (parsing) and easily interchanged. SGML is very flexible, mobile and versatile.

A comparison of some of the characteristics of HTML, PDF, SGML is done [4]:

Attribute	HTML	PDF	SGML
Portability	Portable : Standard for the Web on all platforms; free browsers	Easily distributed on Web; free viewer available	Portable but Web viewer not yet available for all platforms

File size	Small : ASCII markup + content	Smaller files than Postscript, but still quite large	Small : ASCII markup + content
Hyper linking	Yes	Yes	Yes
Layout	Flexible	Inflexible : tied to numbered, printed page	Flexible
Web features	Support hypertext with tables, lists, interactive forms, graphics, animation and others	Adheres exactly to page layout	User-defined tags and architecture (DTD) allow rich document structure; better suited to longer complex documents with long life-cycle
Cost	Inexpensive for simple application; investment increases with complexity and functionality	Economical	Expensive
Markup	ASCII-based, easy to code, good editing tools. For example : HoTMetal, HotDog or any word processor	Easy to create from PostScript or typeset files or scanned images of printed pages	Complex : requires expertise and expense to set up; expensive tools; steep learning curve
Format	User has limited control over format (fonts, background); printed document is usually large with	Excellent print quality	Screen/print display controlled through (multiple) style sheets; user can modify; no two-

	lots of white space		column styles as yet
Speed	Small files load quickly in browser	Large files is slow to download (but new streaming plug-in speeds viewing somewhat)	Small files, but loading takes a little while to interpret "support" files (DTD, style sheet, navigator)
Tagging	Limited tag set that defines appearance rather than structure/content	No hierarchical structure/document elements	Open-ended, user-defined tag set; structure/database-oriented; context-sensitive
Access	Cross-platform, standard-based core syntax, but proprietary extensions lead to lack of stability; not user defined	Proprietary	Cross-platform; long-term viability through open standard

Table 2.2 Comparison of some of the characteristics of HTML, PDF, SGML

2.6.2 Archiving

Archiving and preservation of original digital documents is an important issue. The Association of Research Libraries (ARL) standards for archiving digital materials at this time specify one of two formats: HTML or PDF. No one has come up with satisfactory provisions for handling multimedia.

The concepts permanence and durability are well established in the literature of preservation. Permanence refers to those qualities and characteristics of an

information carrier which enable it to resist deterioration through aging, assuming that it is not subject to severe environmental conditions over a period of time or a cataclysmic disaster. The definition of permanence implies that deterioration will take place whether or not the information carrier is handled and its information accessed. Durability refers to those qualities and characteristics which enable an information carrier to resist damage through the process of use.

The following factors need to be considered in deciding on archival procedures:

- Making and maintaining tape backups of electronic theses should be standard operating procedure; however, magnetic media are not secure from data loss.
- Keeping an archival paper copy, at least for an initial period, could ease the transition to electronic format for those stakeholders who would resist abandoning print completely.
- Writing the dissertations to CD-ROMs might provide more-secure long-term preservation.

For long-term preservation, which brings up the question of technological obsolescence, one must think of not only the content but also the medium. The two common methods of digital preservation are refreshing and migration. Refreshing data (copying the content periodically, such as from an old tape to a new one) is not likely to solve the long-term problem. For complex files, critical functionality in the original file can eventually be lost. The technique of migration, however, holds more promise. It is possible to migrate older files through newer stages of technology when the new system has been designed to emulate the old so that it can accept the old files.

Migration preserves the data and form of the original. Most current hardware and software designers take care to ensure that new versions will accept older files.

Digital surrogates of physical material have the advantage of being infinitely replicable without degradation. Common practice, when it comes to archiving, has been to record these digital surrogates onto storage media such as CD-ROM, which soon becomes redundant. The development of the World Wide Web has created the opportunity, not only to replicate content, but also to distribute it widely.

A preservation strategy using the World Wide Web, as a 'keeping place', appears to have been overlooked as a viable long-term repository for digital material. The problem of hardware obsolescence is simply pushed out to a single point at the server level where backward compatibility and incremental upgrade is the norm. As an evolving medium for the distribution of digital work, its paradigms are transitional and may well be flexible enough to carry digital content into the future.

UMI still considers microfilming to be its dependable archival medium. Virginia Tech considers UMI to be its "emergency" backup, since it does not keep an archival paper copy itself. Virginia Tech runs a script (similar to a macro) that automatically generates an e-mail message to UMI whenever a new dissertation has been mounted, giving the URL, author, and title. UMI then downloads whenever it chooses, prints, microfilms, and adds the file to its own digital database. However, Virginia Tech hopes to address the problem of long-term preservation and access of electronic theses by periodically writing electronic theses to CD-ROMs for security back-ups and possibly longer term preservation

2.7 Features of Electronic Theses and Dissertations

Features	Australian Digital Theses Program	Concordia University ETD Collection	M.I.T Theses Online
Browse by: 1. Author 2. Department 3. Partner institution 4. Year 5. Title	 X 	 X X 	 X X
Search by: 1. Title 2. Author 3. Subject words 4. Department 5. Date/Year 6. URL 7. Keyword 8. Abstract 9. Words 10. Phases 11. Advisor name	 X X X X X X X	 X	 X X
Advanced search Operator : 1. AND 2. OR 3. AND NOT	 X		 X X
Links to other sites	X	X	X
Help	X		X
Feedback	X		X
Delivery format	PDF	PDF	GIF
View thesis : 1. Chapter by chapter 2. Full documentation	 X	 X	 X X

Features	NCSU Libraries	West Virginia University	Virginia Tech
Browse by: 1. Author 2. Department 3. Partner institution 4. Year 5. Title	X	X X	X X
Search by: 1. Title 2. Author 3. Subject words 4. Department 5. Date/Year 6. URL 7. Keyword 8. Abstract 9. Words 10. Phases 11. Advisor name	X	X X X	X X
Advanced search Operator : 1. AND 2. OR 3. AND NOT			X
Links to other sites	X	X	X
Help	X		X
Feedback	X	X	X
Delivery format	PDF	PDF	PDF
View thesis : 1. Chapter by chapter 2. Full documentation	X	X	X

Features	University Of Waterloo E-thesis Project	Worcester Polytechnic Institute (WPI) Electronic Theses Collection	University of Michigan
Browse by: 1. Author 2. Department 3. Partner institution 4. Year 5. Title		X X	X X
Search by: 1. Title 2. Author 3. Subject words 4. Department 5. Date/Year 6. URL 7. Keyword 8. Abstract 9. Words 10. Phases 11. Advisor name	X X X X X	X X X X	X X X X X X
Advanced search Operator : 1. AND 2. OR 3. AND NOT		X X X	
Links to other sites	X	X	
Help			
Feedback			
Delivery format	PDF	PDF	SGML/HTML
View thesis : 1. Chapter by chapter 2. Full documentation	X	X	X

Table 2.3 Features of Electronic Theses and Dissertations

CHAPTER 3 SYSTEM ANALYSIS

3.1 Introduction

System analysis is an important phase because it is used to identify the system's functional and non-functional requirements as well as the software and hardware requirements that are needed to support the identified functions. The emphasis throughout this phase is to produce a user-oriented description of what exactly the system will do. All users requirements for the proposed system are documented well.

3.2 Fact Finding Techniques

Information on several areas such as electronic theses system are obtained by a process called requirements determination and this process also known as fact finding. There are a few data-gathering methods used to define requirements. The fact-finding techniques used include interviews, the Internet, research and observation.

3.2.1 Interviews

An information-gathering interview is a directed conversation with a specific purpose that uses a question-and-answer format. Interviews have been conducted with professionals from the library science field to get the opinions of the interviewees and their feelings about the electronic theses digital library system. Some of the interviewees include Mr. Ling Tek Chaw, Mr. Phang Keat Keong, Mr. Teh Kang Hai and Prof. Madya Zainab Awang Ngah. They are lecturers from the Faculty of Computer Science and Information Technology.

3.2.2 The Internet

The Internet is a very effective way of obtaining information. There are many sites available in the Internet that provide information on electronic publishing, electronic thesis submissions and electronic theses systems. Besides, there are many samples of electronic theses systems from various universities all around the world that can be used as guidelines in determining the system's functional requirements.

3.2.3 Research

Research is another important fact-finding technique, which involve reviewing books and articles published in the journals that contain relevant information. A clearer perception of an electronic theses digital library system is known well after the literature review. Besides, studies were done on the current e-theses web applications. E-mails had been sent to the publishers to obtain valuable advices and up to date information.

3.3 Functional Requirements

Functional requirements are functions or subsystems that are mandatory to the system. The absence of the functional requirements will make the whole system incomplete. The following describes the functional requirements for the Digital Library of Theses.

3.3.1 Information retrieval (Search)

One of the important modules of the Digital Library of Theses is the search function. Therefore, the system should provide many types of search criteria and search strategies to enable users to choose from to perform a search. The system should also provide the basic search criteria such as searching by Author, Title, Keywords, Word(s) in the abstract, Year and Faculty. The search engine in this system is used to search through the database of the server and through the database of the CD-ROMs.

There are two types of search strategies provided by this system :

3.3.1.1 Simple Search

The simple search strategy allows user to search by title, author, keywords, word(s) in abstract, year and faculty for theses. In addition, this system enables users to perform a search for the source code. The objects are search by object name, programming language, word(s) in the object description and keywords. The full source code are search by the title of the project, author, year, word(s) in the object description and keywords.

3.3.1.2 Advanced Search (Boolean Search)

Boolean Search is a combinatorial search that enables professional users to perform a search by using logical connectors. The logical connectors are used to join several fields together to facilitate more accurate and precise searching. The operators provided in the Digital Library of Theses are AND and OR. The AND operator will narrow down the search scope whereas the OR operator will broaden the search scope.

3.3.2 Display

The system should provide two types of display; record listing and full record listing. The record listing displays the entire search results whereas the full record listing shows the full documentation of a particular thesis selected from the record listing of the search results. This system will use PDF as the delivery format of the full record listing.

3.3.3 Sorting

Both the simple and advanced search results can be sorted according to the selected fields such as title, author and year (as selected by the users).

3.3.4 Theses and source code uploading

Students who are involved in thesis writing and system developing are required to submit their theses and source code in the electronic form. Templates are created for the students to fill in the theses' information, author's information, supervisor's name, name of department, name of faculty, year, searchable keywords and abstract. For those students who need to submit their source code of their system, they may submit

the source code in the form of reusable components, objects or full source code. Thus, another template is created for the students to fill in object's name, searchable keywords, programming languages being used and description of the objects. The students can upload the theses and source code to the server. However, only qualified students who are given a unique password can access to the templates for uploading. This is to prevent unauthorized uploading of information to the server. This is to control the security of the system. The types of file uploaded are set to pdf or zip.

3.3.5 Viewing and administering news

This function allows the users to read up on any news or announcements from the library. Besides, the library administrators are allowed to perform administration functions such as adding new records, deleting records and updating records.

3.3.6 Sending and administering feedback

This feature will enable users to send in their suggestions or comments to the library. Library administrators should be allowed to perform administration functions such as viewing all the feedback, deleting feedback and replying the feedback.

3.3.7 Help

This feature provides help to assist users in using all the functions mentioned above including search, theses and source code uploading, access to news and feedback function and theses indexing.

3.3.8 Theses listing

In addition, a thesis listing is provided. Theses are listed by year and by author. Year indexing and author indexing are important to assist users to search for a thesis.

3.4 Non-Functional Requirements

Non-functional requirements define system properties and constraints. Non-functional requirements are as important as functional requirements and sometimes are more critical than functional requirements. The following are the non-functional requirements for the Digital Library of Theses.

- **Response time**

One of the crucial factors in retrieving information using a web-based system and in establishing a connection to the database server is the response time (from the web server and also from the database server). When a user submits a search queries or other request in the Digital Library of Theses, they should not keep waiting for a long time for the results.

- **User friendliness**

The design of the system and its interface should be user friendly and easy to use. The design of all the interfaces in the system should possess the following criteria.

a) Consistent, in terms of screen design and error messages displayed.

b) Accommodation of any level of users; both professional users and novice users.

c) Appropriate error handling with associated pop-up window error messages

d) High degree of understandability and avoid too much using of commands and memorization of events

- **Accurate**

The system should response to the search queries requested by the users accurately. The search results displayed to the users should be accurate.

- **Robustness**

This system should be able to check the input validation before it continues for further data processing to avoid unnecessary disaster. When there is any mistake detected, this system should prompt an error message to indicate the mistake. For example, when a user does not key in value for a mandatory field, an error message should be prompted.

- **Modularity**

Modularity is important for further enhancement and modification as the users' requirements change over time. The system is divided into four modules. These modules are independent and isolated to each other. They may be useful in any of the full or partial programs in other similar applications.

- **Correctness**

Correctness includes traceability, completeness and consistency. The logic of the system must be able to be traced. In this system, comments are written in the scripts. In addition, performance of this system should be consistent and stable.

- **Reliability**

A reliable system should be consistent and functioning well. Thus, this system should run smoothly although there are many web users using the system simultaneously. The system should not produce dangerous or costly failures when it is used in a reasonable manner.

3.5 Consideration Of Programming Language Technologies

The consideration of programming language technologies used in the implementation of the Digital Library of Theses is based on two current popular technologies and they are the Active Platform and the Java Platform.

The Internet has created immense market potential for developers seeking to create component-based applications for the emerging network-centric computing model. Sun Microsystems and its allies, such as IBM and Netscape have long been promoting Java as a cross-platform development environment made for the Internet.

Sun introduced a component framework for delivering apples called Javabeans and Microsoft has responded with ActiveX, part of the recently announced Active Platform development environment, which essentially extends the Windows/OLE desktop to the Net. The standoff between these two technologies is expected to force developers to take sides in an increasingly bitter market battle.

To better understand what these choices are, the following discusses on the Active Platform, Java Platform and a comparison between them. But before going into details about the platforms, an introduction to the Microsoft Common Object Model, Microsoft Distributed Common Object Model and CORBA is presented.

3.5.1 Common Object Model (COM)

3.5.1.1 Overview of COM

The Component Object Model (COM) is an object-oriented architecture for building applications. COM is a platform-independent, distributed, object-oriented, system for creating binary software components that can interact. COM is the foundation technology for Microsoft's OLE (compound documents), ActiveX (internet enabled components), as well as others. COM consists of a well-defined, mature, stable, and freely available specification, as well as a reference implementation, which has been widely tested and adopted worldwide as a de-facto standard. COM is the most widely used component software model in the world. It provides the richest set of integrated services, the widest choice of easy-to-use tools, and the largest set of available applications. In addition, it provides the only currently viable market for reusable, off-the-shelf, client and server components.

COM specifies how to build components that can be dynamically interchanged and provides the standard that components and clients follow to ensure that they can operate together. COM components consist of executable code distributed either as Win32 dynamic link libraries (DLLs) or as executables (EXEs). Components written to COM standard meet all the requirements for a component architecture. The COM components link dynamically by using DLLs. Nevertheless, dynamic linking by itself does not guarantee component architecture and thus the components must be encapsulated.

COM components announce their existence in a standard way. Using COM's publication scheme, clients can dynamically find the components they need to use. COM components are very useful in providing object-oriented APIs or services to other applications as well as efficient for building language-independent component libraries from which applications can be rapidly built.

When discussing about COM, they are a few terms that need further explanation :

- **Component**

A unique of executable code that provides functionality. Components are provided by servers which are either .exe, .dll or .ocx files. Servers can be made up of one or more components and components provide the templates from which objects are created. The Component Object Model (COM) specifies how components are created and client applications connect to components. COM also processes requests from client applications to create objects.

- **Object**

A combination of codes and data that can be treated as a unit. An object has a lifetime; it is created and destroyed. COM is a technology that allows objects to interact across process and machine boundaries as easily as objects within a single process interact.

- **Automation**

Part of the COM specification that defines a standard method for creating components and using objects. Automation objects are also known as programmable objects.

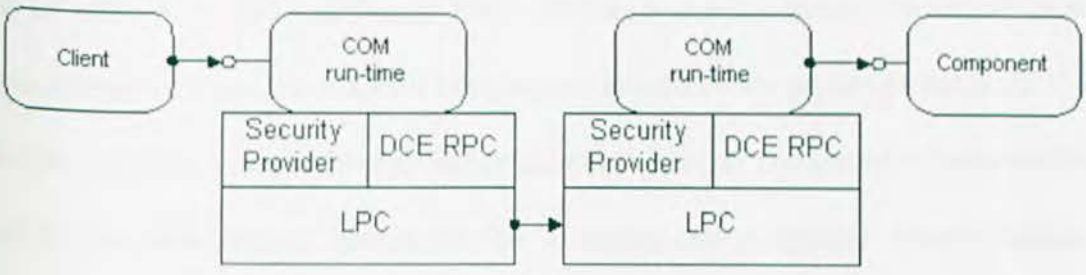


Figure 3.1 COM components in different processes

The only language requirement for COM is that code is generated in a language that can create structures of pointers and, either explicitly or implicitly, call functions through pointers. COM defines the essential nature of a COM object. In general, a software object is made up of a set of data and the functions that manipulate the data. A COM object is one in which access to an object's data is achieved exclusively through one or more sets of related functions. These function sets are called interfaces, and the functions of an interface are called methods. Further, COM requires that the only way to gain access to the methods of an interface is through a pointer to the interface.

Besides specifying the basic binary object standard, COM defines certain basic interfaces that provide functions common to all COM-based technologies. It also provides a small number of API functions that all components require. COM has now expanded its scope to define how objects work together over a distributed environment, and added security features to ensure system and component integrity.

To the developer of client applications, an object takes input and provides output, but its internal working are not available to the developer. Developers only need to

understand how to use the functionality offered. This functionality is offered through one or more published interfaces. These interfaces are the means by which client applications communicate with the component. Interfaces are groups of functions that provide connection points through which clients and server components communicate and it also standardized access to the methods and properties (functionalities) available from the servers. Further, there is a contract between the component author and the client developer that ensures consistent access to functionality. Finally, they structure the access so that servers are easier to use.

3.5.1.2 COM Servers Types : In Process or Out Process

A COM server is any object that provides services to clients. These services are in the form of implementations of COM interfaces that can be called by any client who is able to get a pointer to one of the interfaces on the server object. There are two main types of servers, in-process and out-of-process. A server is referred to as "in-process" (a .dll) when its code executes in the same process space as the client whereas an "out-of-process" server (an .exe) means it runs in another process on the same machine or in another process on a remote machine. In addition, COM provides a mechanism that allows an in-process server (a DLL) to run in a surrogate EXE process to gain its advantages, such as being able to run the process on a remote machine. These three types of servers are called :

- **In-Process Server**

It is a server that can be loaded into the client's process space and serves "in-process objects" and these are implemented as "dynamic link libraries" or DLLs. Efficient exchange of information can be provided by the communication between a client and

an in-process server component. Calls are made directly between the client and the server component because both of them share the same address space.

- **Local Server**

It is a server that runs in a separate process on the same machine as the client. It also serves "local objects". This type of server is another application of its own thus defining a separate process. This specification uses the terms "EXE" or "executable to describe an application that runs on its own process.

- **Remote Server**

A server that runs on a separate machine and therefore always runs in another process as well to server "remote objects". Remote servers may be implemented in either DLLs or EXEs, if a remote server is implemented in a DLL, a surrogate process will be created for it on the remote machine.

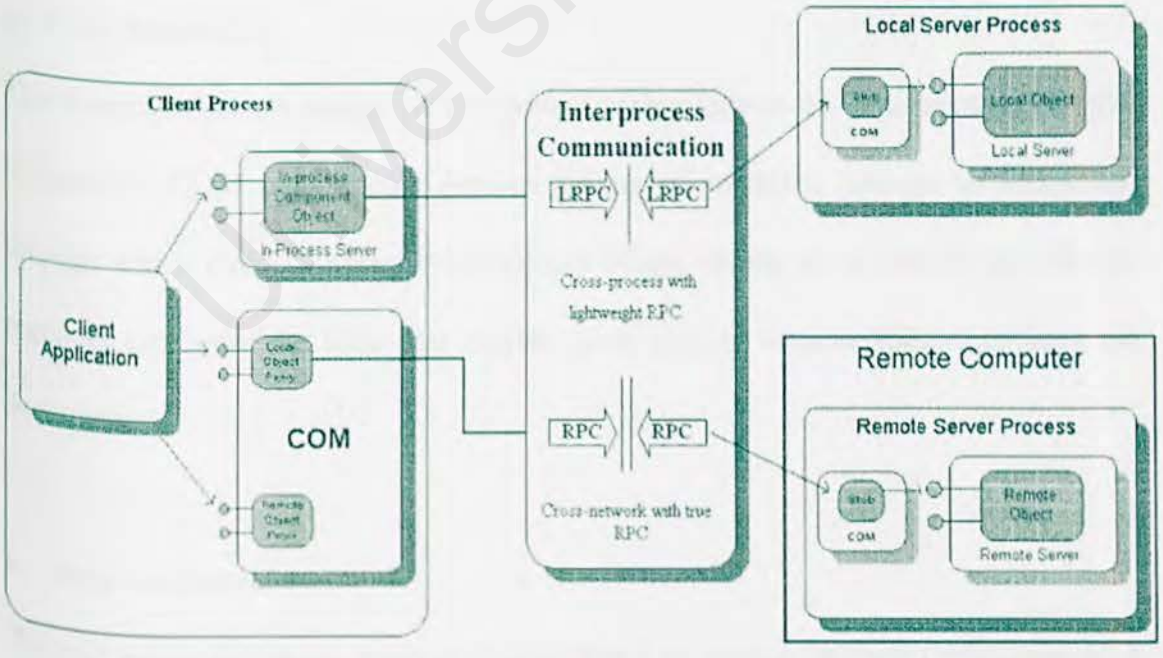


Figure 3.2 Clients always call in-process code; component objects are always called by in-process code. COM provides the underlying transparent RPC.

3.5.1.3 Advantages of COM

Component technology helps to create applications more efficiently because it provides advantages such as code compatibility, reusability and version control. The following discuss on these advantages :

- **Binary Compatibility and Cross-Platform Development**

The first major advantage of component technology is code compatibility across computing platforms. Since becoming an industry-wide standard, COM-compliant client applications and servers are independent of the development language and operating system. COM ensures complete binary compatibility between the client that is developed in Microsoft Visual Basic and the components developed in Microsoft Visual C++ or Microsoft Visual J++ or components running on a Macintosh or a UNIX server.

- **Code Reusability**

The second major advantage of component technology is an enhancement of DLL technology. COM components expose groups of methods, known as interfaces, through which clients interact with objects. Many clients for a variety or different purposes can reuse the code that creates those objects because these interfaces are documented.

- **Version Control**

The last major advantage resolves issues related to version control. COM-compliant components are said to be self-versioning. This means that new functionality can be added to a component (by adding or changing the implementation of an interface)

without affecting clients that already use the component. Functionality is not lost when components are upgraded; it is always enhanced or upgraded.

3.5.2 Distributed Common Object Model (DCOM)

3.5.2.1 Overview of DCOM

DCOM (Distributed Component Object Model) is a set of Microsoft concepts and program interfaces in which client program object can request services from server program objects on other computers in a network. DCOM is a protocol that enables software components to communicate directly over a network in a reliable, secure, and efficient manner. DCOM is based on the Component Object Model (Component Object Model), which provides a set of interfaces allowing clients and servers to communicate within the same computer (that is running Windows 95 or a later version). The Distributed Component Object Model (DCOM) extends the Component Object Model (COM) to support communication among objects on different computers—on a local area network (LAN), a wide area network (WAN), or even the Internet. Previously called "Network OLE," DCOM is designed for use across multiple network transports, including Internet protocols such as HTTP. DCOM is based on the Open Software Foundation's DCE-RPC spec and will work with both Java applets and ActiveX® components through its use of the Component Object Model (COM).

DCOM can also work on a network within an enterprise or on other networks besides the public Internet. It uses TCP/IP and Hypertext Transfer Protocol. DCOM comes as part of NT 4.0 and is a free upgrade for Windows 95. DCOM is or soon will be

available on all major UNIX platforms and on IBM's large server products. DCOM replaces OLE Remote Automation.

DCOM is generally equivalent to the Common Object Request Broker Architecture (Common Object Request Broker Architecture) in terms of providing a set of distributed services. DCOM is Microsoft's approach to a network-wide environment for program and data objects.

Because DCOM is a seamless evolution of COM, the world's leading component technology, one can take advantage of their existing investment in COM-based applications, components, tools, and knowledge to move into the world of standards-based distributed computing. DCOM will then handles low-level details of network protocols to provide transparency in developing distributed applications.

In today's operating systems, processes are shielded from each other. A client that needs to communicate with a component in another process cannot call the component directly, but has to use some form of inter-process communication provided by the operating system. COM provides this communication in a completely transparent fashion: it intercepts calls from the client and forwards them to the component in another process.

When client and component reside on different machines, DCOM simply replaces the local inter-process communication with a network protocol. Neither the client nor the component is aware that the wire that connects them has just become a little longer. Figure 3.3 shows the overall DCOM architecture: The COM run-time provides object-

oriented services to clients and components and uses RPC and the security provider to generate standard network packets that conform to the DCOM wire-protocol standard.

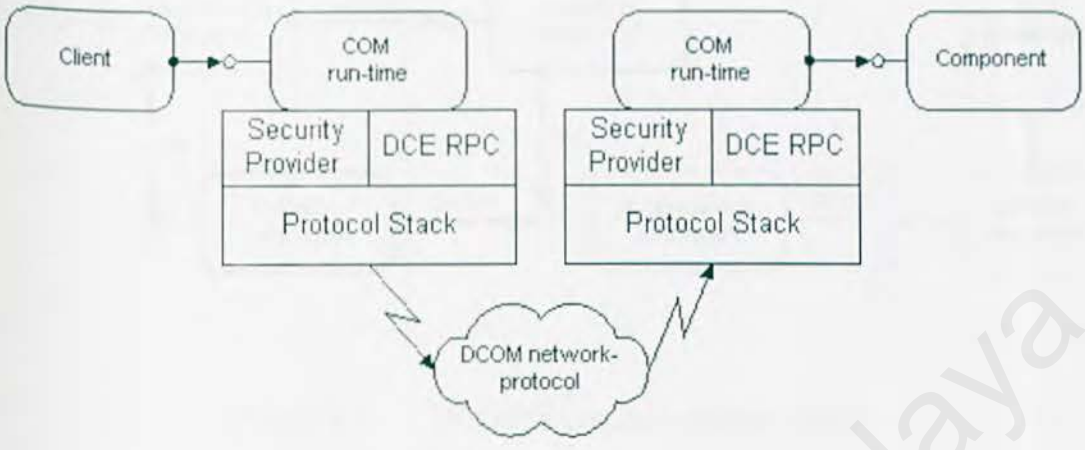


Figure 3.3 DCOM Architecture

DCOM directly and transparently takes advantage of any existing investment in COM components and tools. Any component that is developed as part of a distributed application is a candidate for future reuse. With DCOM's location independence, the application can combine related components into machines that are "close" to each other onto a single machine or even into the same process. Even if a larger number of small components implement the functionality of a bigger logical module, they can still interact efficiently among each other. Components can run on the machine where it makes most sense: user interface and validation on or close to the client, database-intensive business rules on the server close to the database.

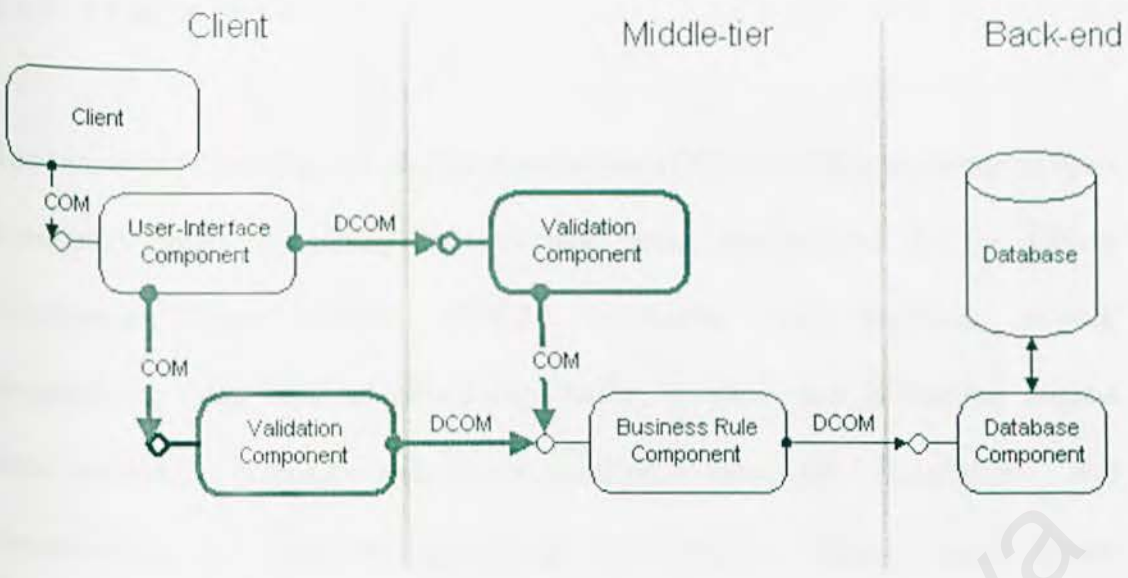


Figure 3.4 DCOM's Location Independence

A common issue during the design and implementation of a distributed application is the choice of the language or tool for a given component. Language choice is typically a trade-off between development cost, available expertise, and performance. As an extension of COM, DCOM is completely language-independent. With DCOM's location independence, the application can combine related components into machines that are "close" to each other onto a single machine or even into the same process. Even if a larger number of small components implement the functionality of a bigger logical module, they can still interact efficiently among each other. Components can run on the machine where it makes most sense: user interface and validation on or close to the client, database-intensive business rules on the server close to the database.

3.5.3 CORBA

The Common Object Request Broker Architecture (CORBA) [4] is an emerging open distributed object computing infrastructure being standardized by the Object Management Group (OMG). CORBA automates many common network programming tasks such as object registration, location, and activation; request demultiplexing; framing and error-handling; parameter marshalling and demarshalling; and operation dispatching. The Common Object Request Broker Architecture (CORBA) is the Object Management Group's answer to the need for interoperability among the rapidly proliferating number of hardware and software products available today. Simply stated, CORBA allows applications to communicate with one another no matter where they are located or who has designed them. CORBA 1.1 was introduced in 1991 by Object Management Group (OMG) and defined the Interface Definition Language (IDL) and the Application Programming Interfaces (API) that enable client/server object interaction within a specific implementation of an Object Request Broker (ORB). CORBA 2.0, adopted in December of 1994, defines true interoperability by specifying how ORBs from different vendors can interoperate.

The (ORB) is the middleware that establishes the client-server relationships between objects. Using an ORB, a client can transparently invoke a method on a server object, which can be on the same machine or across a network. The ORB intercepts the call and is responsible for finding an object that can implement the request, pass it the parameters, invoke its method, and return the results. The client does not have to be aware of where the object is located, its programming language, its operating system,

or any other system aspects that are not part of an object's interface. In so doing, the ORB provides interoperability between applications on different machines in heterogeneous distributed environments and seamlessly interconnects multiple object systems.

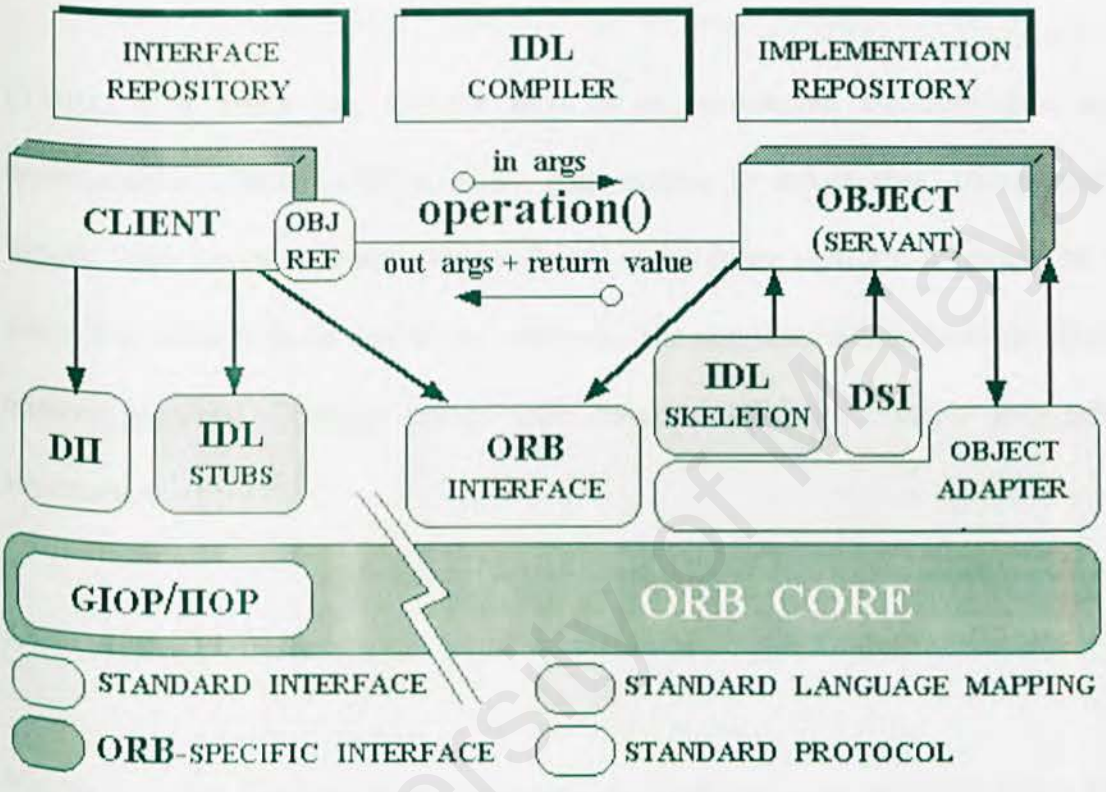


Figure 3.5 CORBA ORB Architecture

In fielding typical client/server applications, developers use their own design or a recognized standard to define the protocol to be used between the devices. Protocol definition depends on the implementation language, network transport and a dozen other factors. ORBs simplify this process. With an ORB, the protocol is defined through the application interfaces via a single implementation language-independent specification, the IDL. And ORBs provide flexibility. They let programmers choose the most appropriate operating system, execution environment and even programming

language to use for each component of a system under construction. More importantly, they allow the integration of existing components. In an ORB-based solution, developers simply model the legacy component using the same IDL they use for creating new objects, then write "wrapper" code that translates between the standardized bus and the legacy interfaces.

CORBA is a single step on the road to object-oriented standardization and interoperability. With CORBA, users gain access to information transparently, without them having to know what software or hardware platform it resides on or where it is located on an enterprises' network. The communications heart of object-oriented systems, CORBA brings true interoperability to today's computing environment.

3.5.4 Active Platform

Microsoft Active Platform is an integrated, comprehensive set of client and server development technologies that make it easy for developers to integrate the connectivity of the Internet with the power of the personal computer. Based on Microsoft Corp.'s leading standards-based implementation of HTML, open scripting, and component architecture, the Active Platform allows developers to use the tools and components they know today to build powerful applications easily for the Internet and intranets. The Active Platform includes client technologies that run in an Active Client browser, server technologies that run on the Windows NT® Server network operating system and Microsoft Internet Information Server (IIS), and ActiveX™

technologies that tie client and server components together across the network and enable component-based applications to work seamlessly.

Active Platform not only brings uniformity to the server side for Web-enabled applications, it also encompasses the client side with the same open architecture to deliver a cohesive means of displaying content residing on Active Platform servers. There are three main parts to the Active Platform that make it possible to create, deploy, and use powerful Web-enabled applications for the Internet and intranets. These include the following:

- **ActiveX Technologies**

ActiveX technologies refers to the tools and standards that work as the glue to bring dynamic, interactive content to your Web site. ActiveX technologies are a set of software standards, components, and tools that work on different operating systems and heterogeneous networks to enable the Active Platform.

- **Active Desktop**

The Active Desktop is the client component of the Active Platform environment. The Active Desktop enables the creation of applications that run on the client systems under a multitude of operating systems and hardware platforms.

The Active Desktop provides developers with a means of writing applications to a common interface to ensure the capability to run on multiple operating systems and hardware platforms. Active Desktop includes support for language-independent scripting, dynamic HTML, system services, and ActiveX component technology.

- **Active Server Pages**

The Active Server Pages is a high-level component that takes advantage of the scaleable, high-performance capabilities of Windows NT Server to provide developers with a rich environment for creating server-side Web applications. Active Server Pages includes full support for Windows NT Server system services, database access, transaction processing, and message queuing.

The main advantage of Active platform is that it allows developers to move onto the Web without sacrificing existing investments in training and components. It also means that ActiveX controls can be easily integrated into a wide variety of existing Window applications such as Microsoft Word, Excel and even BackOffice. The Active Clients is cross-platform which is found built into Microsoft Internet Explorer 3.0 and above; it can be part of the client/server application through ActiveX. For Windows environment, Active platform provides a sound and scalable server-side application development platform.

The core technology of Active platform is the COM. If used in a network with a directory and additional support, COM becomes the Distributed COM (DCOM). The most important thing when writing a program to run in the ActiveX environment is a component, a self-sufficient program that can be run anywhere in the ActiveX network (currently a network consisting of Windows and Macintosh systems). This component is known as an ActiveX Control.

3.5.4.1 ActiveX Controls

ActiveX® controls are among the many types of components that use COM technologies to provide interoperability with other types of COM components and services. ActiveX control is a component program object that can be re-used by many application programs within a computer or among computers in a network. ActiveX controls can be downloaded as small programs or animations for Web pages, but they can also be used for commonly-needed task by an application program in the latest Windows and Macintosh environments.

ActiveX controls are the third version of OLE controls (OCX), providing a number of enhancements specifically designed to facilitate distribution of components over high-latency networks and to provide integration of controls into Web browsers. These enhancements include features such as incremental rendering and code signing, to allow users to identify the authors of controls before allowing them to execute.

3.5.5 Java Platform

The Java Platform is a new software platform different from many other platforms like Microsoft Windows, Macintosh, OS/2, UNIX and NetWare; it is designed to deliver and run highly interactive, dynamic, and secure applets and applications on networked computer system. The Java Platform sits on top of these operating system. Applications written in the Java language compile to architecture neutral bytecodes for the Java Virtual Machine, rather than normal machine code. A Java interpreter then executes the translated code. The primary advantage of this approach is that it allows a Java application to run on any system, as long as that system implements the

Java Virtual Machine. This portability is possible because at the core of the Java Platform is the Java Virtual Machine.

While each underlying platform has its own implementation of the Java Virtual Machine, there is only one virtual machine specification. Due to this factor, the Java Platform can provide a standard, uniform programming interface to applets and applications on any hardware. The Java platform is therefore ideal for Internet, where one program should be capable of running on any computer in the world. The Java platform is designed to provide the "Write Once, Run Anywhere" capability.

Developers can write object-oriented, multithreaded, dynamically linked applications using the Java Language. The platform has built-in security, exception handling and automatic garbage collection. Just-In-Time compilers are available to speed up execution by converting Java bytecodes into machine language. From within the Java Language, developers can also write and call native methods in C, C++ or another language, compiled to a specific underlying operating system for speed or special functionality.

On the Java Platform developers can create two different kinds of programs: applets and applications. The primary difference between them are: an applet requires a network to run while an application does not. While applets and applications for the most part have the same access to a wide range of language capabilities, applets are restricted from having read or write access to any file system except the server from which they came. This is because an applet can be potentially downloaded from an untrusted source. This constraint will be relaxed when mechanism is implemented so

that applets can be marked with digital signatures, allowing the end-user to be assured that it has been downloaded unaltered from a trusted source.

3.5.5.1 JavaBeans

JavaBeans is a portable, platform-independent component model written in the Java programming language, developed in collaboration with industry leaders. It enables developers to write reusable components once and run them anywhere -- benefiting from the platform-independent power of Java technology. JavaBeans acts as a Bridge between proprietary component models and provides a seamless and powerful means for developers to build components that run in ActiveX container applications.

JavaBeans components, or Beans, are reusable software components that can be manipulated visually in a builder tool. Beans can be combined to create traditional applications, or their smaller web-oriented brethren, applets. In addition, applets can be designed to work as reusable Beans.

JavaBeans is a complete component model. It supports the standard component architecture features of properties, events, methods, and persistence. In addition, JavaBeans provides support for introspection (to allow automatic analysis of a JavaBeans component) and customization (to make it easy to configure a JavaBeans component). JavaBeans brings the extraordinary power of the Java platform to component development, offering the ideal environment for a developer who wants to extend the concept of reusable component development beyond one platform and one architecture to embrace every platform and every architecture in the industry.

JavaBeans does not add any security features to the Java platform. Rather, JavaBeans components have full access to the broad range of security features that are part of the Java platform. JavaBeans components can be used to build a range of different kinds of solutions from full-fledged Java desktop applications to web-based Applets.

3.5.6 Comparison of ActiveX and Java

Criteria	ActiveX	JavaBeans
Platforms	ActiveX is platform specific and compiles into Windows bytecode. Microsoft has promised to port ActiveX to other platforms.	JavaBeans operates as a cross-platform environment and compiles into an architecturally neutral "J-Code".
Security	ActiveX has serious problems that may prove difficult to fix. Its controls have access to your computer's operating system, files and applications. Security depends on trusted servers and digital signatures.	JavaBeans works with the Java Virtual Machine implementation, which prevents applets from accessing files or applications. The language was designed from the start with security in mind. Security is not foolproof, but holes are generally discovered and patched quickly.
Speed	ActiveX has advantage because it is optimized for the Windows platform and arrives over the Internet as Window bytecode, making	JavaBeans is comparatively slow. It arrives in an architecturally neutral state and must be interpreted by the Java Virtual Machine

	it faster than JavaBeans. ActiveX component can run as native code and thus it is processed faster.	before it can be executed.
Optimization	ActiveX is optimized for Windows. Microsoft simply adapted its existing desktop component architecture, OLE for networks. ActiveX controls, however, are already built into Windows 95 and Window NT operating systems as well as several applications; this will allow ActiveX controls to work with Windows programs.	JavaBeans is optimized for the Internet. Java is a component architecture that was designed for the Internet and through cooperation with the OpenDoc standard, will move to the desktop.

Table 3.1 Comparison of ActiveX and JavaBeans

3.6 Consideration of programming languages

3.6.1 Active Server Page (ASP)

An Active Server Page (ASP) is an HTML page that includes one or more scripts (small embedded programs) that are processed on a Microsoft Web server before the page is sent to the user. An ASP is somewhat similar to a server-side include or a common gateway interface (CGI) application in that all involve programs that run on the server, usually tailoring a page for the user. Typically, the script in the Web page at the server uses input received as the result of the user's request for the page to access data from a database and then builds or customizes the page on the fly before sending it to the requestor.

ASP is a feature of the Microsoft Internet Information Server (IIS), but, since the server-side script is just building a regular HTML page, it can be delivered to almost any browser. An ASP file can be created by including a script written in VBScript or JScript in an HTML file or by using ActiveX Data Objects (ADO) program statements in the HTML file. The HTML file is named with the ".asp" file suffix. Microsoft recommends the use of the server-side ASP rather than a client-side script, where there is actually a choice, because the server-side script will result in an easily displayable HTML page. Client-side scripts (for example, with JavaScript) may not work as intended on older browsers.

ASP is considered as a glue technology, which binds together various server-based system to help in building interactive web page. It is able to interact with almost any

existing dynamic web page technology such as CGI (Common Gateway Interface), ISAPI (Internet Server Application Programming Interface) and scripts. In addition, ASP is suitable for building multi-tier Internet and Intranet applications. It also supports client-server running programming. It is able to spread processing load between client and server by implementing and integrating client-side processing as well as server side processing.

3.6.1.1 VBScript

Microsoft Visual Basic Scripting Edition (VBScript) is a subset of Microsoft Visual Basic Language. VBScript does not include features that are normally outside the scope of scripting, such as file access and printing because it was specifically designed to work in browsers. VBScript is supported primarily on Microsoft platforms. VBScript is available as a part of Microsoft Internet Explorer (IE) and Internet Information Server. VBScript does not produce standalone applets but is used to add intelligence and interactivity to HTML documents.

Unlike Visual Basic, VBScript is not used as a designed environment. Users cannot lay out forms and user interfaces by dragging and dropping controls. Instead, VBScript is an after-the-fact language. The placement of ActiveX controls on a web page and how they got there is VBScript's true concern.

If VBScript is used, the following conditions must be taken into consideration :

- **Server script**. If VBScript is used as server script (in Active Server Pages), the server must be IIS or a third-party equivalent.

- **Client script**. If VBScript is used in browser scripts, the browser must be a version of Internet Explorer. Other popular browsers might not support VBScript.

However, the restriction to Internet Explorer for client script is a problem if a public Web site is created because it is impossible to control what browsers' people use. Therefore, using VBScript is practical in client script only in writing for an audience with a known browser, such as a company intranet.

3.6.2 Common Gateway Interface (CGI) Applications

The Common Gateway Interface (CGI) allows Web servers to execute other programs and incorporate their output into the text, graphics, and audio sent to a Web browser. CGI programmers use tools that provide for forms processing, looking up records in a database, or sending e-mail. CGI applications are more like a system utility than full-blown applications; scripts are task-oriented rather than process-oriented. A CGI script has a single job: it initializes, does its job, and then terminates. It is easy to chart data flow and program logic, but CGI has its limitations:

- CGI programs take more time to write and debug; thus produces frequent down time for Web sites
- CGI programming is consistently outperformed by ASP and is five times slower
- CGI Web pages are "non-dynamic", thus continuous changes cannot be made "on the fly."
- CGI is not inherently multi-threaded (ASP is), which limits the number of concurrent users

- CGI uses a greater amount of server resources degrading performance of servers and sites.

3.6.3 Comparison of Programming Languages and Components

Criteria	Common Gateway Interface (CGI)	Active Server Pages (ASP)	Cold Fusion
Operating System Support	UNIX, DOS, Mac, Windows and Others	Windows and Sun Solaris	Windows and Sun Solaris
Hardware requirement	None	None	None
Software requirement	None	None	Cold Fusion Package
Web server compatibility	UNIX, Windows, Netscape and Mac Web servers	Windows web servers and Netscape's web server	Windows web servers and Netscape's web server
Scripting languages	C/C++, Perl, VBScript, JavaScript	VBScript, JavaScript, Perl	Cold Fusion Markup language
Learning issues	Not very easy to learn	Easy to learn	Easiest to learn among the three
Limitation	Resource intensive on server if there are multiple requests	Highly IIS dependant. Compatibility with other web servers	Highly database centric. Limited by CF tags.
Greatest advantage	Platform independence and multi-language support	Rapid application development in a team setting	Designed especially for publishing databases

Costs involved	Development costs	Development costs	Product, license and development costs
Recommendation	Good for almost any application except when there are going to be a large number of hits simultaneously	Good for almost any kind of application. Works especially well in a Windows NT environment	Good for applications accessing databases like a E-Commerce site and handling large number of hits

Table 3.2 Comparison between ASP, CGI and Cold Fusion

3.7 Other Considerations

3.7.1 Microsoft Windows NT Server 4.0

Microsoft Windows NT Server 4.0 is a complete and powerful platform that provides server operating system. It provides the backbone for a complete, organic system, where all elements working together seamlessly [5]. When it is joined with other Windows NT-related products, including the BackOffice® family of applications and Windows NT Workstation, Windows NT Server provides the foundation for a powerful and well-integrated environment. The introduction of new management tools in Windows NT Server 4.0 and the Option Pack has provided great assistance in setting up Web Sites, managing the content and analyzing usage patterns for improvement. It serves as platform to publish and share information in a secure way over Intranet and Internet.

3.7.2 Internet Information Server 4.0

Internet Information Server (IIS) is Windows NT 4.0 built-in web server. With IIS 4.0, Microsoft introduces a new paradigm to the web-transactional applications.

Transactions are the plumbing that now makes it possible to run real business applications with rapid development, easy scalability and ATOMIC reliability. IIS 4.0 brings together the power of Windows NT Server, with the best of client/server development, and the ubiquity of the Internet to create the first Web platform for true distributed applications. The server services provided by IIS are WWW, FTP and Gopher.

3.8 Conclusion

The programming language technology chosen is based on the Active Platform. Active platform can provide dynamic web-page generation. All changes made to the current web-page can be reflected on the very same page, unlike the static web-page (HTML) implementation.

ActiveX (which is based on COM/DCOM) is chosen for creation of business object or component (as a middle-tier to segregate all the business rules from the server). ActiveX component including the ActiveX Controls and ActiveX Documents can be easily created using the popular Microsoft Visual Basic Ver 5.0. ActiveX is optimized for Windows platform and thus can be executed faster than the JavaBeans. There are also many third party tools that are available for development as compared to JavaBeans. Since ActiveX is based on the COM/DCOM standard, this COM/DCOM standard is chosen for development of this system.

Since the programming language technology chosen is based on the ActiveX and Microsoft NT Server works well with ActiveX, therefore Microsoft Windows NT Server is chosen as the development platform. The Web server used in the implementation of the Digital Library of Theses is the Internet Information Server (IIS) Ver 4.0, which is web server for the Windows NT platform that enables developers to publish information on a corporate intranet or on the Internet. Microsoft Internet Information Server (IIS) Ver 4.0 is chosen because it supports ASP. Internet Information Server transmits information by using the Hypertext Transfer Protocol (HTTP). Internet Information Server can also be configured to provide File Transfer

Protocol (FTP) and gopher services. The FTP service enables users to transfer files to and from your Web site. The gopher service uses a menu-driven protocol for locating documents.

There are several reasons as to why the Windows NT platform is chosen here for the development and implementation of the Digital Library of Theses

- NT comes integrated with web server, IIS and thus can be used efficiently
- NT Server is the only true multipurpose operating system. It combines the performance of file and print servers and the power of UNIX application servers with the ease of use of Windows.
- NT offers ease of administration or maintenance where administrators can learn, use and manage one system with powerful file and print services plus robust and reliable application services.
- NT Server is interoperable with other systems such as NetWare; it integrates with legacy systems while offering a smooth migration. NT supports up to 32 processors.
- With the available resources during the development of the Digital Library of Theses, that is, a personal computer with Pentium Pro processor, the NT serves as the most suitable platform.
- Though NT may not be as robust, as scalable and as reliable as the UNIX platform but it is sufficient for middle scale development and use.
- During development, the Active Platform was only available in NT (and not in UNIX, but Microsoft has already planned to enhance the platform support for Active Platform which is also now available in Macintosh platform).

3.8.3 Choice of Programming Language

It is important to understand the functional and non-functional requirements for the system while considering the programming language to be used for development. There are several criterias that form the foundation of choosing the appropriate programming language.

- The programming language should be able to cater for dynamic web-based programming.
- The programming language must be able to support database communication
- The programming language must be able to support the implementation of three-tier client/server architecture.
- The programming language must be able to support ease of building a graphical user interface as the end-users have close interaction with the system.

With the above criterias, the selection of programming language for development is Active Server Pages (ASP). ASP is chosen as a web-page development tool because it can provides dynamic web page development. It is tightly integrated with Windows NT Server and Microsoft Internet Information Server Ver 4.0.

Besides, ASP can provides various benefits as mentioned below :

- Complements server side scripting
- Compile-free development
- Extensible environment
- Easy to learn

- Easily leverage existing investments/skills

VBScript is chosen as the main scripting language for the development of this system because it is easier to learn compared to JavaScript. It also communicates well with the ActiveX components created.

3.9 Hardware and Software Requirements

3.9.3 Server Side

A minimum of 32 MB of RAM is suggested for a Windows NT Server System, but 64 MB of RAM is more appropriate. It is recommended that implementation of SQL Server in a production environment on anything greater than a Pentium 200. The faster the processor speed and more memory the web server has, the better the performance will be. Network Interface Card (NIC) and network connection with recommended bandwidth at 10Mbps or more, a hard disk at least 5 GB of storage and others standard computer peripherals are also required.

To host and run the system, the server computer needs to have various supporting software installed.

Software/ Components	Description
Windows NT Server 4.0	Network Operating System
Internet Information Server	Web-server service
Active Server Pages	Server Scripting Engine
Microsoft SQL Server 7.0	RDBMS for data warehousing
Microsoft Internet Explorer	Precondition for ASP installation
Adobe Acrobat 4.0	Create PDF files from any application

Table 3.3 Server Side Software Requirements

3.9.4 Client Side

The client hardware requirements are quite minimal as long as it has a reasonable amount of RAM and a reasonable quality dial-up connection line. The recommended configurations are:

1. At least 16 Megabytes of RAM.
2. A minimum 150 MB of hard disk storage.
3. Network connection through existing network configuration.
4. Other standard computer peripherals.

Clients need to have the basic Microsoft Operating System such as Windows 3x or Windows 95 as a basic requirement to install other software to support the system. As for compatibility reason Microsoft products are recommended. The client software requirements fall on the browser used by users. It requires a system that can run Internet Explorer 4.0 and above or any other browser that support VBScript. Adobe Acrobat Reader 3.01 also required.

CHAPTER 4 SYSTEM DESIGN

4.1 Introduction

System design is the phase where the requirements of the system are translated into system characteristics, that is a model representation of an entity that will be built later. The goal of the system design effort is the design of an information system that will be effective, reliable and maintainable. To be effective the system must satisfy the defined requirements and meet the specified constraints. The most important factor is that the design must conform to the end-user needs.

During the design of this system, the following considerations have been taken into account:

- **End-User Consideration**

The design of the interface and screen must conform to user needs and to also anticipate future end-user needs. The one factor that weights most heavily while deciding for a design is the effect of it on user. It would be futile to build a powerful system whereby the users find it difficult to use.

- **Data Considerations**

Data should enter the system where and when it occurs, because unnecessary delays can lead to incorrect or lost data. When data enters the system, it should be validated and verified immediately to avoid introducing errors to the system. Besides ensuring accuracy, redundancy should be avoided, such as when a user checks for his/her status

he/she needs to login for once only, and when he/she performs renewal (or reservation) he/she need not login again.

- **Processing Considerations**

This consideration is to aim for processing simplicity. The focus here is to create or build independent modules that perform a single function to ease understanding and maintainability. Thus the modularity factor is given emphasis.

4.2 Design Technique

The design technique used in the Digital Library of Theses is the stepwise refinement technique. Stepwise refinement is a top-down technique for decomposing a system from high-level specification into more elementary level. It is also known as stepwise program development. This technique was originally described with Wirth and it involves the following designing activities

- Decomposing design decisions to elementary levels
- Isolating design aspects that are not truly interdependent
- Postponing decisions concerning representation details as long as possible
- Carefully demonstrating that each successive step in the refinement process is a faithful expansion of previous steps

The major advantages of using stepwise refinement as a design technique are:

- Top-down decomposition
- Incremental addition of detail

- Postponement of design decisions
- Continual verification of consistency (formally or informally)

Using stepwise refinement, a problem is segmented into small, manageable pieces and the amount of detail that must be dealt with at any particular time is minimized.

4.3 User Interface Design

User Interface Design is a crucial process as the user interface is the only way the user communicates with the system. A difficult to use and an unattractive user interface does not attract users to use them. Regardless of how powerful a system may be, a badly designed user interface will shed users from using it. Therefore it is important that this part of the development be given serious and careful consideration.

4.3.1 Screen Design

The main purposes of a screen are to present information and to assist users in using the system. The following guidelines are therefore considered in the screen design for Digital Library of Theses.

- All screen displays should be attractive, uncluttered and symmetrically balanced. Boxes are used to highlight certain groupings of data elements.
- Information on a single screen should be displayed in a meaningful and logical order

- Screen presentations should be consistent; titles, messages and instructions should all appear at the same general locations. Similarly, be consistent in the use of terms.
- All messages should be explicit and understandable

Figure 4.1 shows the general screen design for the Digital Library of Theses.

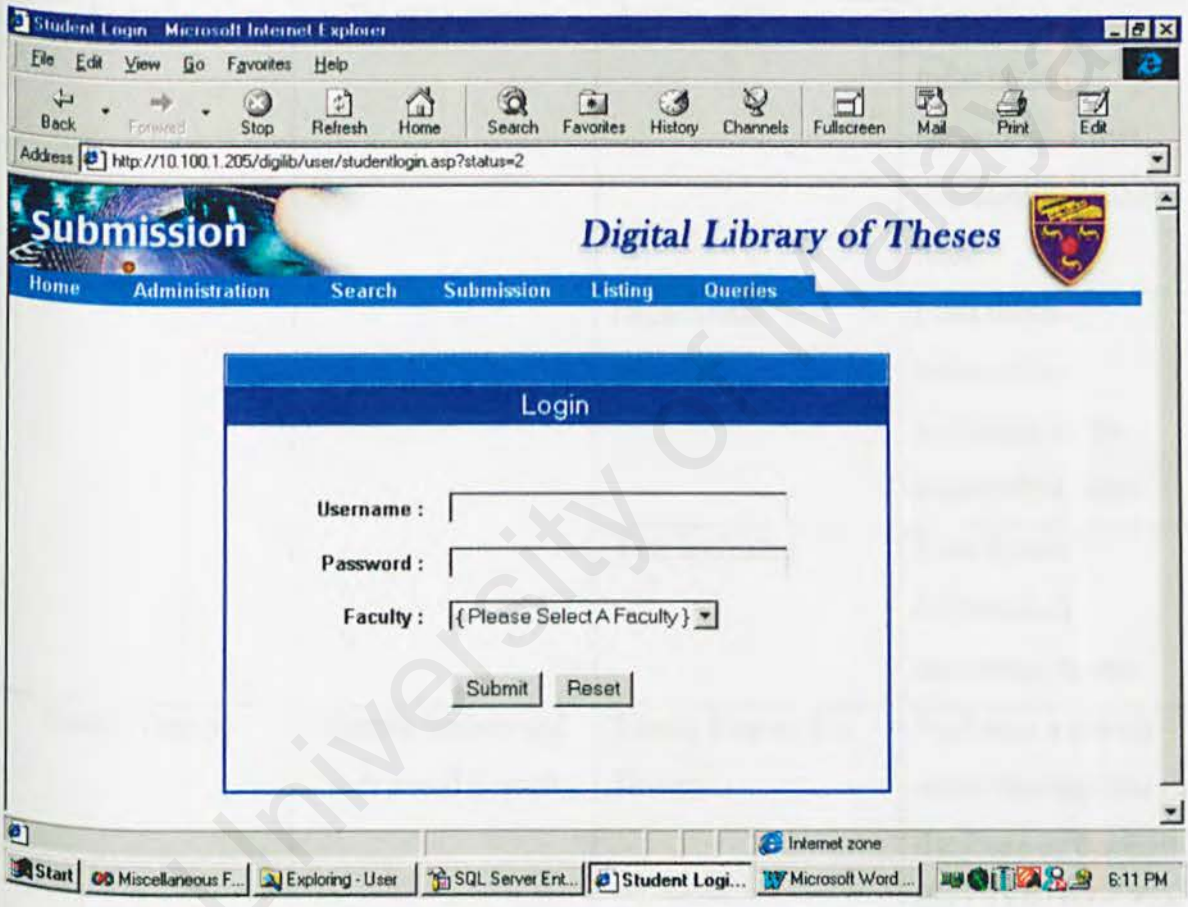


Figure 4.1 Screen Design for Digital Library of Theses

Table 4.1 describes the menu for this system

Module	Menu Item	Sub-Menu Item	Description
	Home		Returns to the default page of the Digital Library of Theses.
Listings	Theses Listings	Author indexing	Lists theses' information according to first character of author's name.
		Department indexing	Lists theses' information according to the department name.
		Year indexing	Lists theses' information according to year.
Search Engine	Simple Search and Advanced Search	Search Engine For Theses	Performs a search either through the database or through the CD-ROMs. The search page contains many search criteria for precise searching.

Search Engine	Simple Search and Advanced Search	Search Engine For Full Source Code	Performs a search through the database. The search page contains many search criteria for precise searching.
		Search Engine For Objects or Components	Performs a search through the database. The search page contains many search criteria for precise searching.
Theses and source code submission	Submission	Theses Submission	Allows the students to upload their theses directly to the server.
		Objects or Components Submission	Allows the students to upload the components or objects directly to the server.
		Full Source Code Submission	Allows the students to upload their full source code directly to the server.

Table 4.1 Client side menu

4.4 Database Design

Database is a collection of data stored in a particular format and accessed through a computer. Database design involves the activity on modeling the structure of a database that will store and maintain the data records. It includes the transformation of users' processing requirement and unordered information into proper functional requirements. At this phase, all the data elements in the information system are identified, an entity-relationship (E-R) diagram is created and the data store design is normalized.

Normalization is a process that identifies and corrects inherent problems and complexities in the database record design. Normalization reduces data redundancies and by extension, helps to eliminate data anomalies that result from those redundancies. Normalization is carried out through a series of stages called normal forms.

In the database design of the Digital Library of Theses, the tables are normalized to the third normal form. The following gives a brief description on the description on the objectives of the first, second and third normal forms used in the database design of the Digital Library of Theses.

- The First Normal Form is to reduce redundancy
- The Second Normal Form is to eliminate partial dependencies
- The Third Normal Form is to eliminate transitive dependencies

These three stages of normalization is sufficient for the database design of the Digital Library of Theses providing an acceptable level of redundancy while at the same time maintaining efficiency.

4.4.1 Data Dictionary

4.4.1.1`Student Table

Field Name	Field Type	Field Size	Allow Null	Description
StudentID	int	4	No	Unique ID for each thesis
FacID	int	4	No	Unique ID for each faculty
DeptID	int	4	No	Unique ID for each department
name	nvarchar	150	No	Student's name
email	nvarchar	80	No	Student's email address
ICnum	numeric	9	No	Student's identity card number
matrix	nvarchar	20	No	Student's matrix card number
SupervisorID	int	4	No	Unique ID for every supervisor
year	nvarchar	20	No	Year
title	nvarchar	200	No	Title of the thesis
keywords	nvarchar	150	No	Searchable keywords for the thesis
abstract	nvarchar	150	No	The filename where the abstract is kept
filename	nvarchar	150	No	The physical path name of the thesis with the file extension

Table 4.2 Student Table

In the database, the student table is named *tblStudent*. This table stores the information for each uploaded thesis. The *StudentID* field is automatically generated when a new thesis is uploaded. The abstract of each thesis is written to different file because of the vast amount of data. Therefore, the *abstract* field is used to point to the physical file where the abstract for each thesis is kept. All the theses are kept in different files. The *filename* field is used to point to the location of the physical files, which is used to store the theses.

4.4.1.2 Full Source Code Table

Field Name	Field Type	Field Size	Allow Null	Description
CodeID	int	4	No	Unique ID for each full source code
name	nvarchar	150	No	Student's name
matrix	nvarchar	20	No	Student's matrix card number
year	nvarchar	20	No	Year
projectTitle	nvarchar	200	No	Title of the source code
keywords	nvarchar	150	No	Searchable keywords for the full source code
description	nvarchar	400	No	Description of the source code
filename	nvarchar	150	No	The physical path name of the source code with the file extension

Table 4.3 Full Source Code Table

In the database, this table is named *tblSourceCode*. This table stores the information for each uploaded full source code. The *CodeID* field is automatically generated when

a new source code is uploaded. The *filename* field is used to point to the location of the physical files, which is used to store the full source code.

4.4.1.3 Object or Component Table

Field Name	Field Type	Field Size	Allow Null	Description
ObjectID	int	4	No	Unique ID for each object or component
name	nvarchar	150	No	Student's name
matrix	nvarchar	20	No	Student's matrix card number
objectName	nvarchar	150	No	Name of the object or component
keywords	nvarchar	150	No	Searchable keywords for the object
description	nvarchar	300	No	Description of the object
filename	nvarchar	80	No	The physical path name of the object with the file extension
proglanguage	nvarchar	50	No	Programming language used for the development of the object or component

Table 4.4 Object or Component Table

In the database, this table is named *tblObject*. This table stores the information for each uploaded object or component. The *ObjectID* field is automatically generated when a new object or component is uploaded. The *filename* field is used to point to the location of the physical files, which is used to store the object or component. The name of the file uploaded is changed according to the object ID

4.5 System Functionality Design

4.5.1 Client/Server Architecture

A client is defined as a requester of services and a server is defined as the provider of services. A single machine can be both a client and a server depending on the software configuration.

As a result of the limitations of file sharing architectures, the client/server architecture emerged. The client/server software architecture is a versatile, message-based and modular infrastructure that is intended to improve usability, flexibility, interoperability, and scalability as compared to centralized, mainframe, time sharing computing. This approach introduced a database server to replace the file server. Using a relational database management system (DBMS), user queries could be answered directly. The client/server architecture reduced network traffic by providing a query response rather than total file transfer. It improves multi-user updating through a GUI front end to a shared database. In client/server architectures, Remote Procedure Calls (RPCs) or standard query language (SQL) statements are typically used to communicate between the client and server [6,7].

4.5.1.1 Three-tier with an application server.

The three-tier application server architecture allocates the main body of an application to run on a shared host rather than in the user system interface client environment. The application server does not drive the GUIs; rather it shares business logic,

computations, and a data retrieval engine. Advantages are that with less software on the client there is less security to worry about, applications are more scalable, and support and installation costs are less on a single server than maintaining each on a desktop client [6]. The application server design should be used when security, scalability, and cost are major considerations [6].

4.5.1.2 Digital Library of Theses Three-Tier Client /Server Architecture

The Digital Library of Theses is implemented using three-tier client-server architecture as shown in Figure 4.2

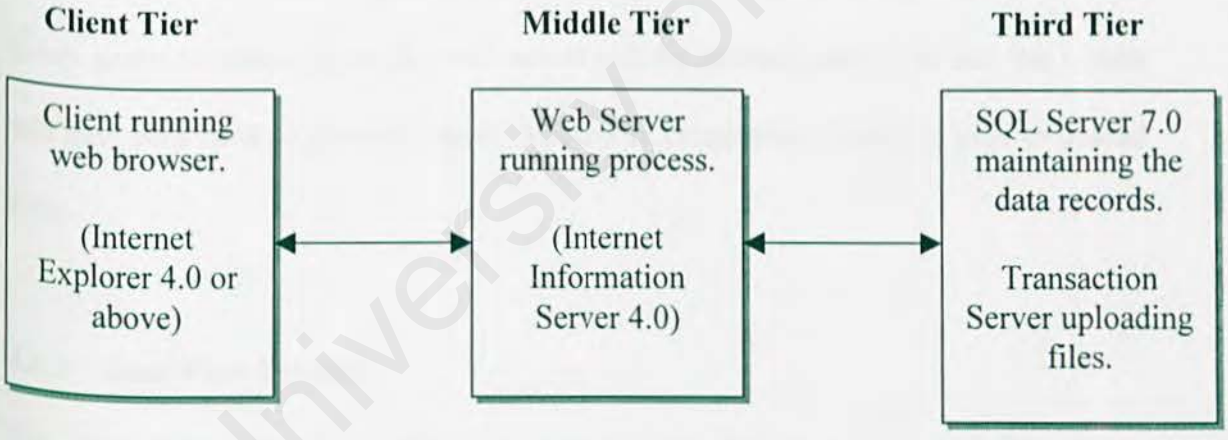


Figure 4.2 Digital Library of Theses Three-Tier Client/Server Architecture

- **Client-tier**

For this system, client tier is constituted of computers with Internet Explorer (4.0 or above). User interfaces are provided for clients to process their application and manipulate their data.

- **Middle-tier**

Middle-tier consists of Internet Information Server 4.0 as the web server. All application programs or files will be resided in the middle-tier (web server). The web server processes the request from the client and then returns required result in web pages format. It will process any data request by linking to database server. For example, authenticating and validating users that login to the server. It is also linked to Microsoft Transaction Server. For example, when clients are uploading files to the web server.

- **Third-tier**

The third tier consists of the Microsoft SQL Server 7.0 as the database server and Microsoft Transaction Server. Microsoft SQL Server 7.0 maintains the data records. Every query requested from the web server will be authenticated first and the results will then pass back to the web server. Microsoft Transaction Server is used to upload files.

4.5.2 Data Flow Diagram

The data flow diagram graphically characterizes data processes and flows in a business system. DFD depicts the overview of system inputs, processes and outputs.

4.5.2.1 Data Flow Diagram for Theses and Source Code Submission

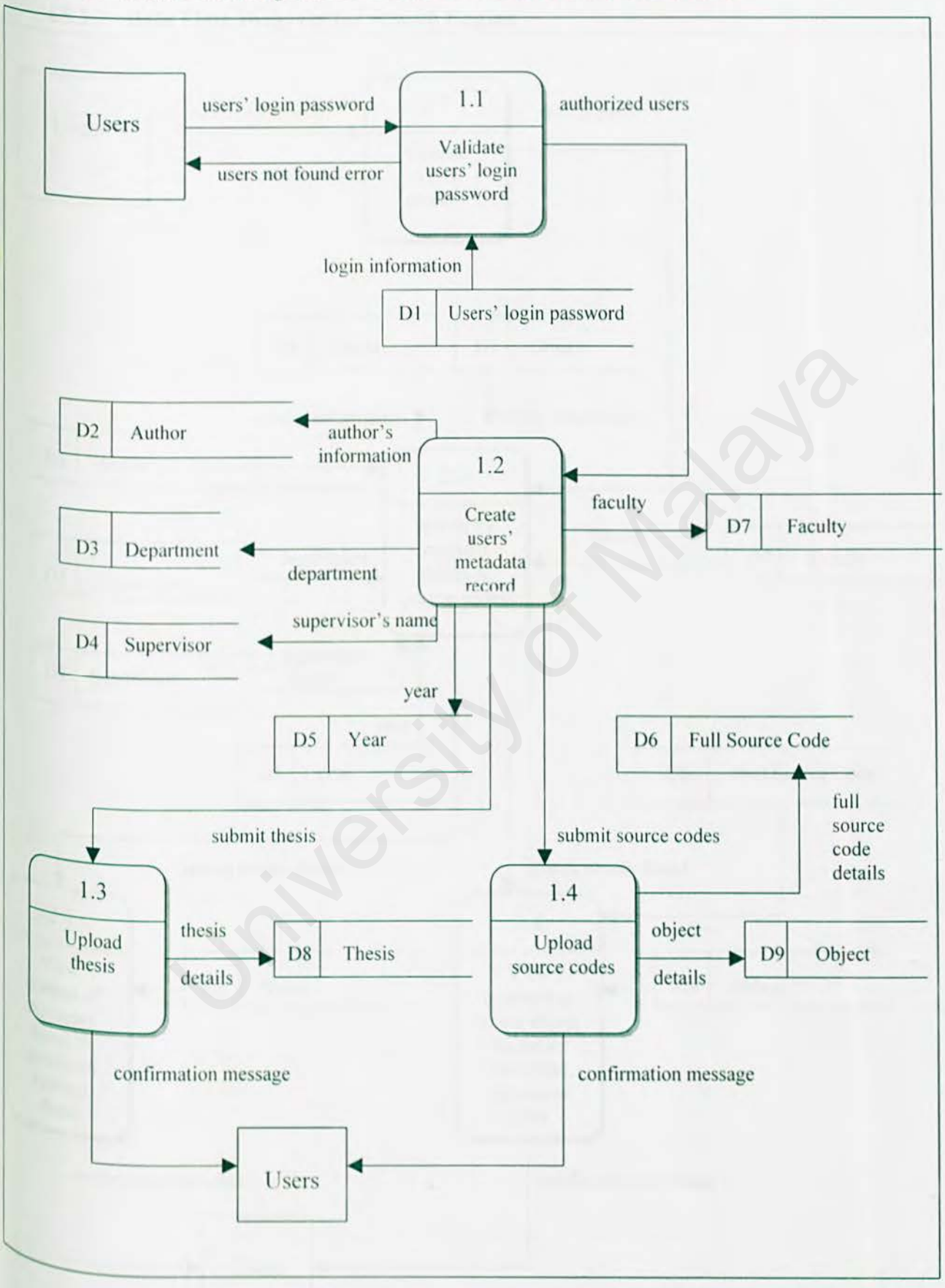


Figure 4.3 Data Flow Diagram for Theses and Source Code Submission

4.5.2.2 Data Flow Diagram for Search Engine

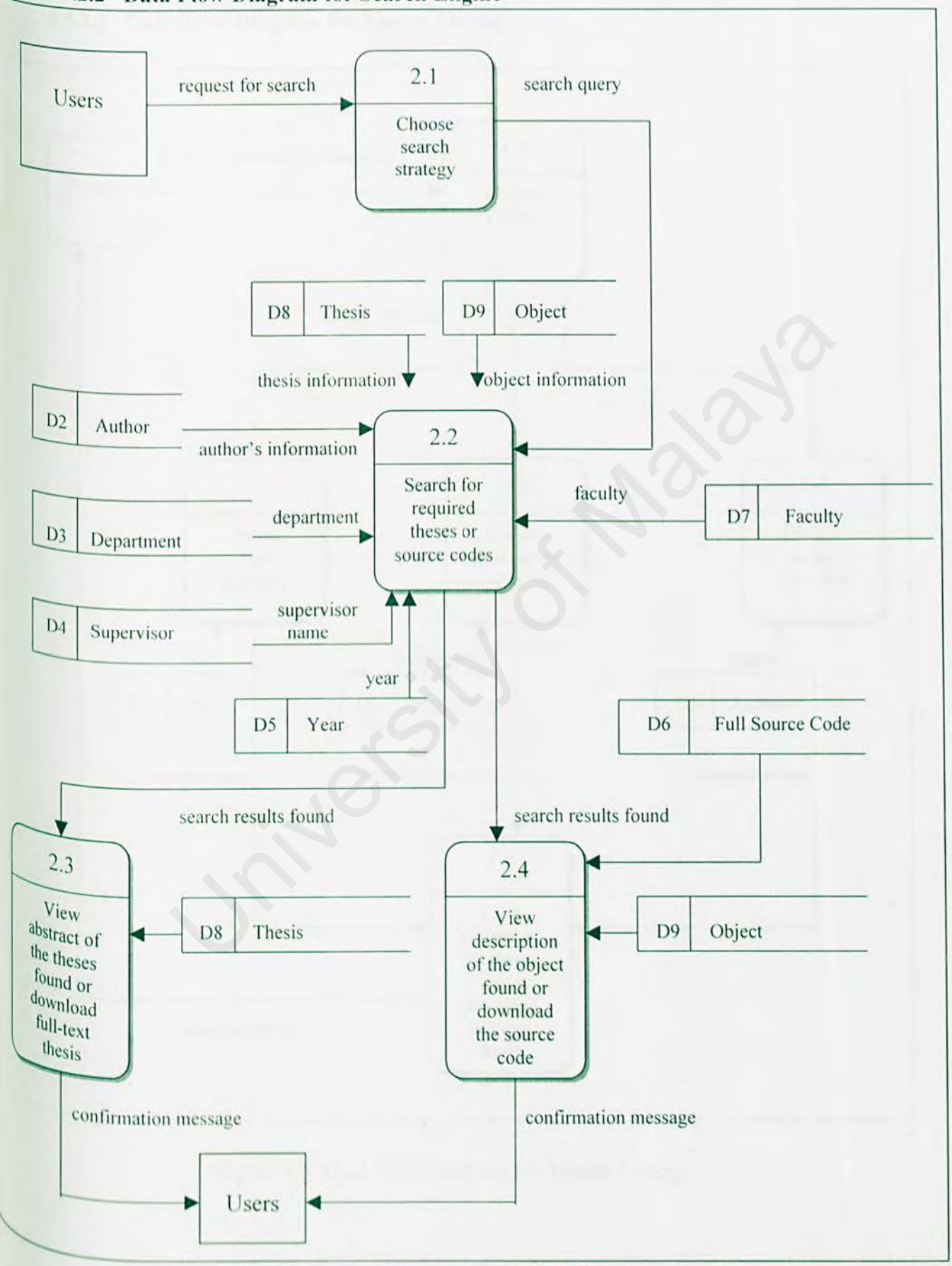


Figure 4.4 Data Flow Diagram for Search Engine

4.5.2.3 Data Flow Diagram for Theses Listing

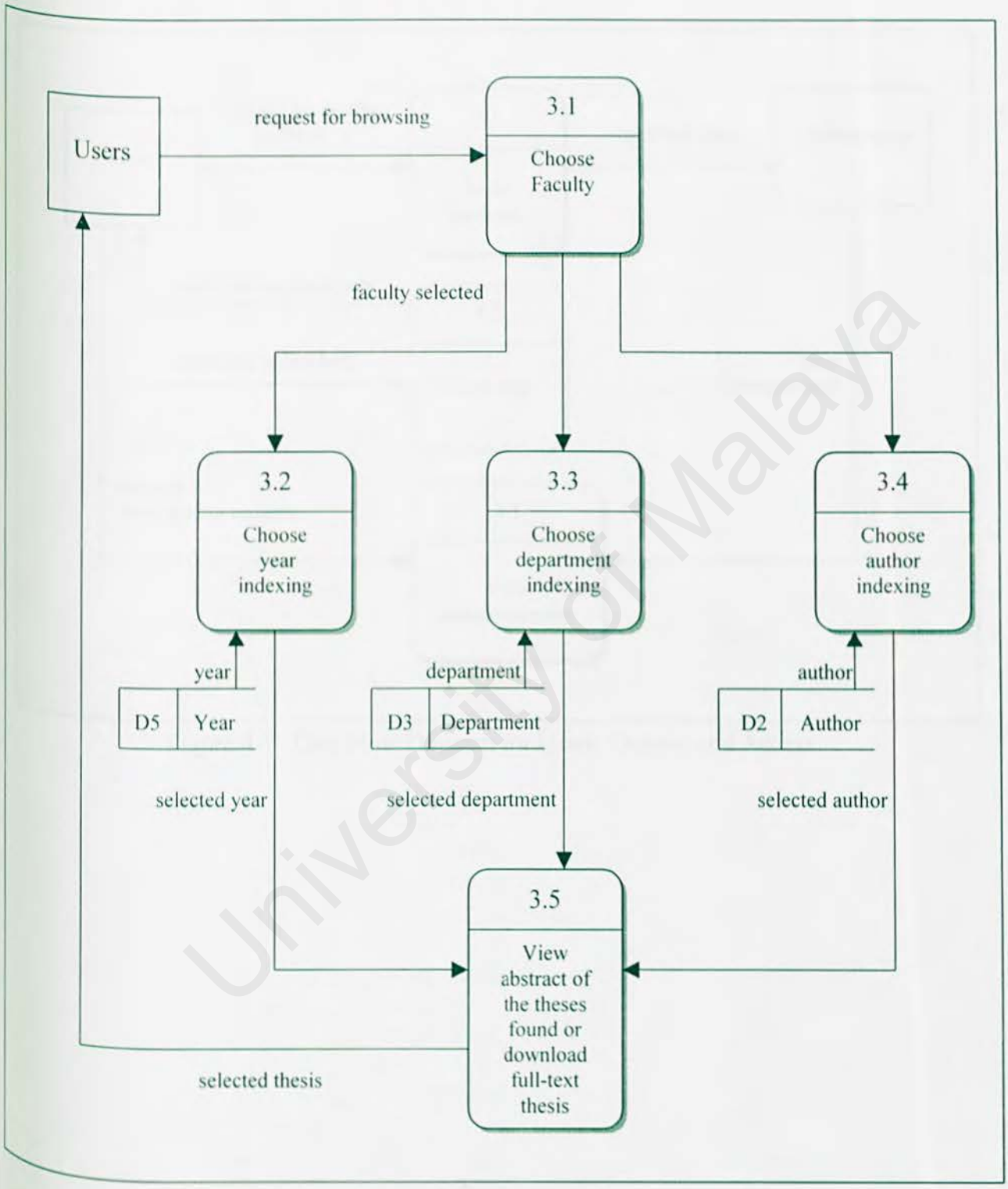


Figure 4.5 Data Flow Diagram for Theses Listing

4.5.2.4 Data Flow Diagram for Users' Queries and Access

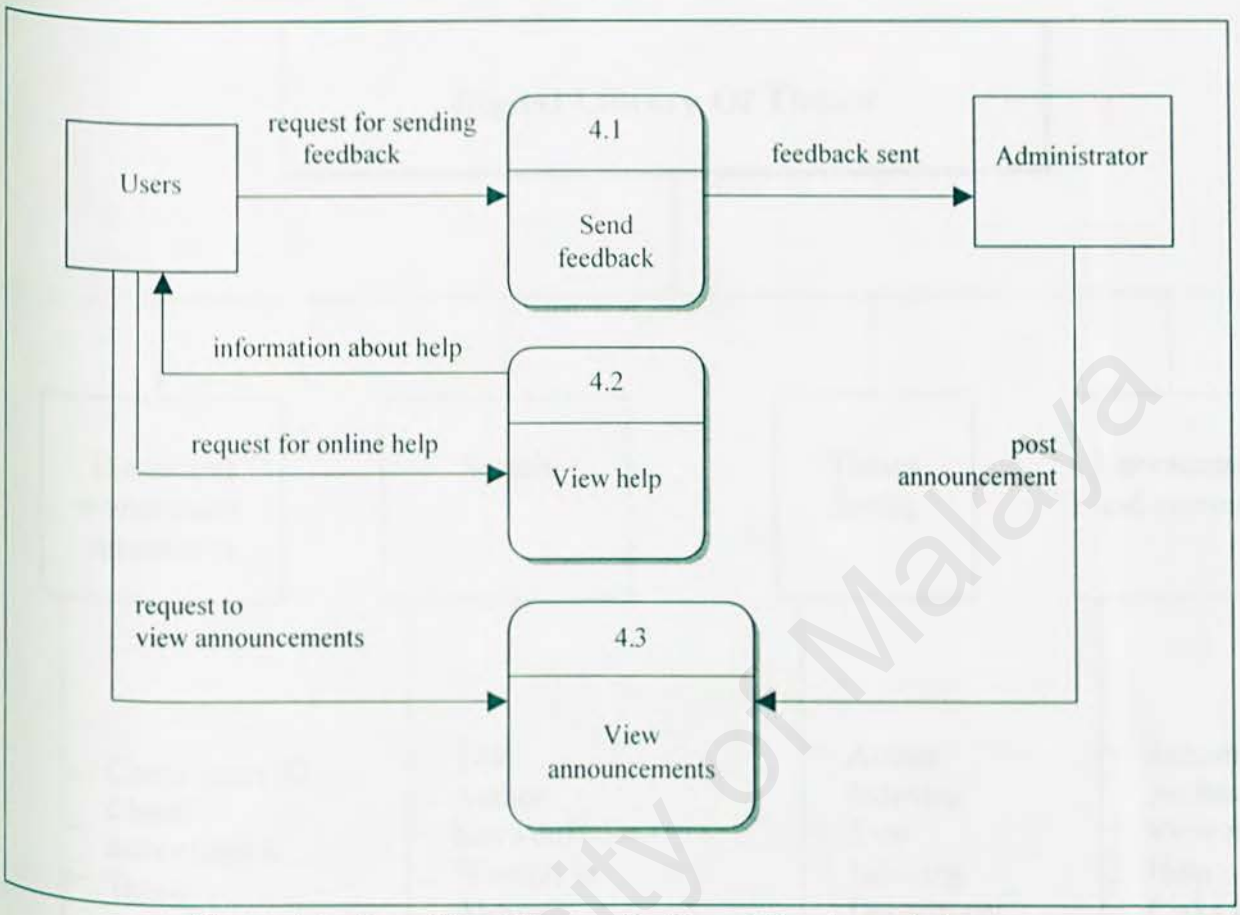


Figure 4.6 Data Flow Diagram for Users' Queries and Access

4.5.3 System Structure Chart

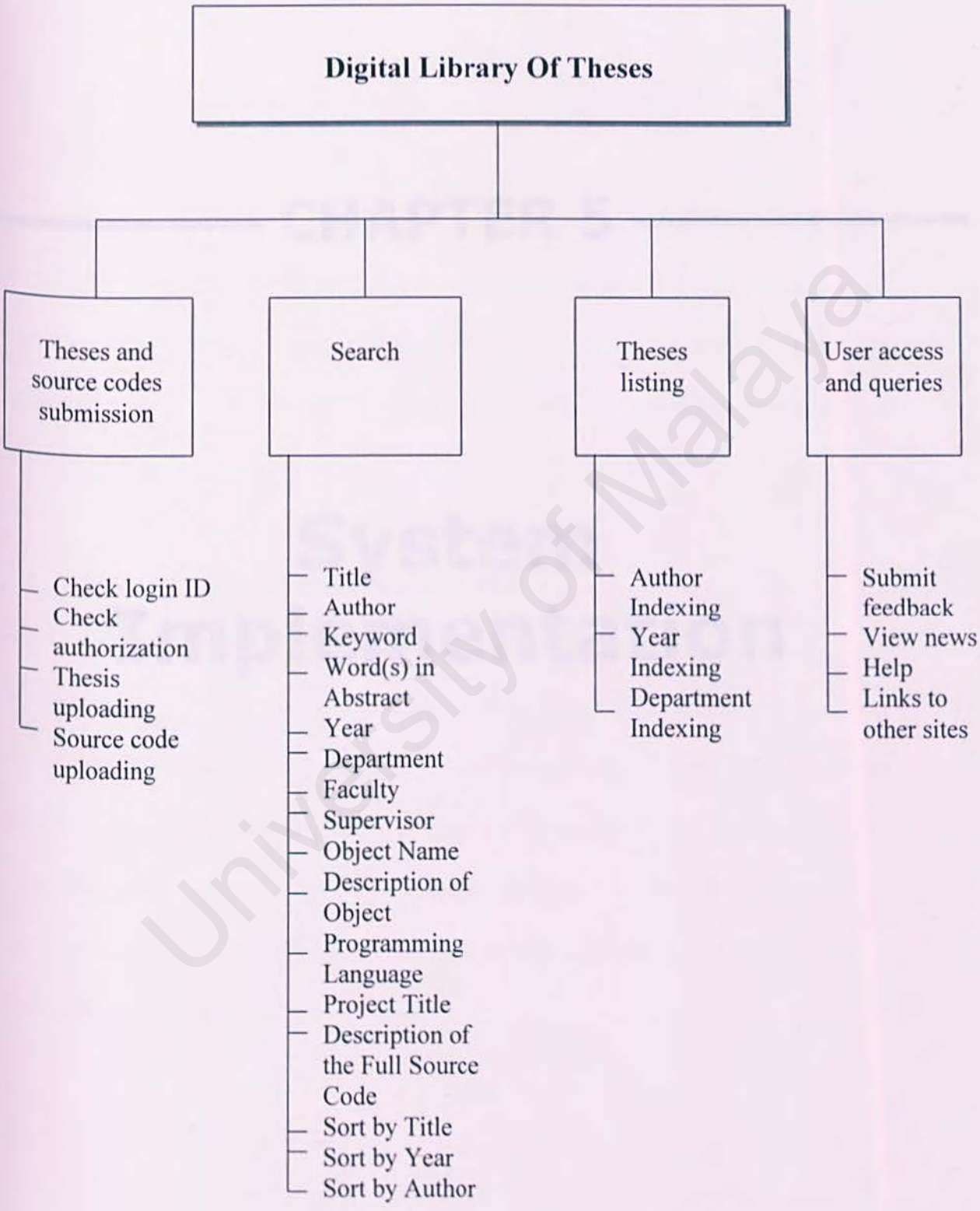


Figure 4.7 System Structure Chart

CHAPTER 5 SYSTEM IMPLEMENTATION

5.1 Introduction

The major sections of system implementation are coding and testing. These 2 sections are discussed in detail in this chapter.

Coding the program is the process of writing program instructions that implement the program design specifications. In other words, it is actually the phase where the software representation produced in the design phase is translated into computer readable form.

5.2 Environment Configuration and Setting

5.2.1 Internet Information Server

The minimum requirement of the web server is Windows NT with Service Pack 4.0. After the Windows NT is installed, Internet Information Server (IIS) has to be installed prior to the Service Pack 4.0. Both the Service Pack and IIS can be found from Windows NT Option Pack 4.0, which is available free from the Microsoft web site. Besides, a file called "*aspSmartUploadUtil.dll*" used for uploading purposes is copied into "*c:\winnt\system32*" directory.

5.2.2 Microsoft SQL Server 7.0

In the Digital Library of Theses, one database has been created. The name of the database is "*E-Theses*".

5.3 Software Tools Used In System Implementation

5.3.1 Development Tools

Software	Description
Microsoft Visual Interdev 1.0	This is very useful in building dynamic web applications using Active Server Pages.
Microsoft SQL Server 7.0	Database server used for storing and manipulating vast amount of data. This database server is used because the system response time and processing time is much faster.
Microsoft FrontPage 2000	Web authoring tool for designing and creating web pages quickly and easily.

Table 5.1 Software Tools Used For System Development

5.3.2 Graphical Design Tool

Software	Description
Adobe Photoshop Version 5.0	Graphical tool used for creating and editing images and icons.

Table 5.2 Software Tool Used For Graphical Design

5.3.3 Documentation Tools

Software	Description
Microsoft Word 2000	Word processor used for writing system documentation.
Microsoft PowerPoint 2000	Tool used for creating presenting slides.

Table 5.3 Software Tools Used For Documentation

5.4 Coding Principles

Throughout coding for the Digital Library of Theses, several principles are followed to ensure quality and proper structure in the code generated and they are:

- **Readability**

Codes should be easily understood. This is an important factor especially when it comes to future enhancing the system by another developer. To cater for this, meaningful variables and labels names have been used; comments are written in each module explaining their functionality and proper indentation are to enhance readability.

- **Maintainability**

Codes should be easily read, corrected and revised. To achieve this, codes should be readable (as explained above), highly cohesive and loosely coupled. Codes that perform functions for one module should be grouped together and try as much as possible to achieve high cohesion and loose coupling.

- **Robustness**

Codes should be robust in terms of handling errors and responding by displaying appropriate error messages and try to avoid system failure. The Digital Library of Theses is developed using the event driven approach, which means that code are executed with respond to provocation of certain events such as the mouse click.

5.4.1 Coding Methodology

The coding methodology approaches that the Digital Library of Theses uses are the top-down approach and the bottom-up approach. These approaches are not only used on individual basis but also as a combination to obtain the benefits from the both techniques.

5.4.1.1 Top-Down Approach

This approach starts by looking at the large picture of the system and then exploding to smaller parts or subsystem. Top-down approach allows the higher-level modules to be coded first before the lower level modules.

This method ensures that the important or core modules of the system to be developed and tested first.

Some of the advantages using this approach are:

- Avoiding the chaos of attempting to code a system all at once
- Prevents the developer from getting so mired in the detail that they loose track of what the system is suppose to do

5.4.1.2 Bottom-Up Approach

In contrast with the top-down approach, the bottom-up approach starts coding at the lower level modules before the higher-level modules are constructed. The higher-level

module acts as an empty shell that calls these lower level modules. The completed lower level module will then be integrated with the newly completed higher-level module.

5.4.2 Client Side Coding

5.4.2.1 Theses Listing

Filename	Description
default.asp	Displays all the faculties. This is the default web page.
dataStore.asp	Establishes connection with the database server.
list(year).asp	Displays the total number of theses according to years and allows the users to view the collection of theses for a particular year.
listing(year).asp	Displays all the theses for a particular year.
list(author).asp	List the theses according to the first alphabet of the author's name.
listing(author).asp	Displays all the theses according to the first alphabet of the author's name.
list(dept).asp	Displays the total number of theses according to departments and allows the users to view the collection of theses for a particular department.
listing(dept).asp	Displays all the theses for a particular department.

Table 5.4 Client Side Theses Listing Coding

5.4.2.2 Theses, Source Code and Components Submission

Filename	Description
studentlogin.asp	Provides form for users to login.
checklogin.asp	Provides checking for the matching username and password entered by the users.
logout.asp	Closes all the sessions and connections. Redirect users to the default web page.
submissionForm(thesis).asp	Provides form for users to upload their theses.
submit(thesis).asp	Uploads the theses to the server.
submissionForm(full source code).asp	Provides form for users to upload their full source code.
submit(full source code).asp	Uploads the full source code to the server.
submissionForm(object).asp	Provides form for users to upload their objects or components.
submit(object).asp	Uploads the objects or components to the server.
registered.asp	Displays message to inform the users that they have successfully uploaded their theses, full source code or objects to the server.

Table 5.5 Client Side Theses, Source Code and Components Submission Coding

5.4.2.3 Search Engine

Filename	Description
simple search (theses).asp	Allows users to search the theses with options.
simple search (object).asp	Allows users to search the objects with options.

simple search (full source code).asp	Allows users to search the full source code with options.
advance search (theses).asp	Allows users to search the theses with Boolean combination for two different fields.
advance search (object).asp	Allows users to search the objects with Boolean combination for two different fields.
advance search (full source code).asp	Allows users to search the full source code with Boolean combination for two different fields.
search1(theses).asp	Display simple search results for theses.
search1(object).asp	Display simple search results for objects.
search1(full source code).asp	Display simple search results for full source code.
search2(theses).asp	Display advanced search results for theses.
search2(object).asp	Display advanced search results for objects.
search2(full source code).asp	Display advanced search results for full source code.
download.asp	Allows users to download the full source code, objects or components.
abstract.asp	Allows users to view the abstract of the thesis.
description(object).asp	Allows users to view the description of the objects or components.
description(full source code).asp	Allows users to view the description of the full source code.

Table 5.6 Client Side Search Engine Coding

5.5 Testing

Testing is a critical element of software quality assurance and it is the final and ultimate review of specification, design and coding. Testing is the process of determining whether a program or a system performs the desired processing. Testing can only shows that the software defects are present. A program must be thoroughly tested to ensure it functions correctly before the program processes actual data and produces information that users will rely on. Testing is a verification and validation process. Verification refers to the set of activities that ensure that the software correctly implements a specific function. Validation refers to a different set of activities that ensure that the software, which has been built, is traceable to customer requirements.

The 3 main objectives of testing are:

- To uncover error in the software whether logical or syntax errors
- To ensure that software functions according too specification and requirements have been met
- Ensure software reliability and quality meaning the software will return what is expected, if all conditions are true

5.5.1 Testing Techniques

Two main testing techniques are used namely the White Box Testing and the Black Box Testing.

5.5.1.1 White-Box Testing

White-Box testing also known as glass-box testing, this testing technique uses the control structure of the procedural design to derive test cases. This method ensures that

- All independent paths within a module have been exercised at least once
- Exercise all logical decisions on their true and false side
- Execute all loops at their boundaries and within their operational bounds
- Exercising internal data structures to ensure their validity

The white-box testing generally ensures that all detail and often left unnoticed errors are taken care off. These errors though may not even occur during processing, may still pop up on a regular basis. Furthermore another frequent problem in coding is the typographical errors. These errors happen randomly and at times are difficult to detect. Using this approach (white-box testing) will far more likely to uncover these bugs.

Thus, the one important point here that has to be given consideration is the coverage of the white-box testing. They are six types of code coverage in white box testing. They are

- **Segment Coverage**

  Every segment of the code between control structures is supposed to execute at least once.

- **Branch Coverage**

  Every branch at every possible direction is executed at least once.

- **Compound Condition Coverage**

  To achieve compound condition coverage, every condition within each decision must be evaluated with every combination of true and false outcomes at some point during test execution. Every possible combination is tested based on the truth table.

- **Data Flow Testing**

  Data flow testing is meant for reflecting dependencies, which are mainly caused by sequences of data manipulation.

- **Loop Testing**

  This type of testing is related to looping. Loop testing is difficult to test when dependencies exist among the loops or between a loop and the code it contains.

- **Basis Path and Path Testing**

  Each independent path throughout the code is taken at a predetermined order. To achieve path coverage, all permutations of paths must be executed. When dependencies appears in the code, each path where dependency appears exists must be tested.

5.5.1.2 Black-Box Testing

Black box testing is also called functional testing. This method focuses on the functional requirements of the software. It ensures that a given set of input will fully exercise all functional requirements of the software. This technique is used to demonstrate that software functions are operational, that input is properly accepted, and the output is correctly produced. It complements white-box testing and is likely to uncover a different class of errors.

Test cases used in black box testing are derived from the specifications of the software. When doing black box testing, functionality is taken into consideration rather than implementation, whereby the program internals are not given serious attention. The program is viewed as a mapping of points from an input space to an output space. Tester will deliberately do some mistakes or enter values that are out of acceptable range to test the system reaction and to determine whether the system will prompt out the appropriate error. Input is provided and output is verified manually to check for accuracy.

The main objectives of black-box testing is to uncover error in incorrect or missing functions, interface errors, errors in data structures of external database access, performance errors and initialization and terminating errors

Unlike white-box testing, which is performed early in the testing process, black box testing tends to be applied during later stages of testing.

5.5.2 Testing Strategies

A software testing strategy should be flexible enough to test all parts of the system. The strategies used in the Digital Library of Theses are unit testing, integration testing and system testing.

5.5.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of the software design namely the module. It is the initial testing stage for the completion of each module. The modules are tested independently of one another to ensure their correct operation.

The objective of unit testing is to identify and eliminate both execution errors, which are errors that cause the program to abnormally terminate, and logic errors, which are errors in the accuracy and completeness of a program's processing. The unit test is normally white-box oriented and the step can be conducted in parallel for multiple modules.

In the Digital Library of Theses, modules that make up the system are mostly form modules, which perform a specific function. Each of these form modules contains many sub functions. Therefore, unit testing was conducted on each sub function first and followed by the module itself.

Some of the unit aspects taken into account are as follows:

- **Module Interface**

This test is carried out to ensure that information properly flows into and out of the program unit under test. Test of data flow across a module interface are required before any test is initiated.

- **Boundary Conditions**

This test is to ensure that the module operates properly at boundaries set up to limit or restrict processing.

- **Local Data Structure**

This test is to ensure that data stored temporarily maintains its integrity during all steps in an algorithms execution.

- **Independent Path**

This test is to ensure all independent paths in a module has been executed at least once in a module.

- **Error Handling Paths**

This test is to ensure that the error-handler of the system is able to handle all expected and unexpected errors that pop up while using the system.

Test Case 1:

After uploading a file, make sure that the file is in the correct destination and the *filename* field in the database points to the same destination. In this system, all the thesis and source code are kept in a different folder. Therefore, the folder in the server is manually checked.

Test Case 2:

After uploading a file, make sure that the filename for each thesis or source code was named properly. Therefore, the filename in the server was manually checked.

Test Case 3:

Test whether the chosen thesis could be read using the Acrobat Reader. Test whether the chosen objects or the chosen full source code could be downloaded from the server.

Test Case 4:

Test whether an appropriate error messages being displayed if a user entered the wrong data in the form field. This was done by purposely making mistakes when using the system.

5.5.2.2 Integration Testing

Integration testing is a systematic technique for testing two or more programs together that depends on one another. Integration testing addresses the issues associated with dual problems of verification and program construction. Integration testing will

usually uncover errors associated with interfacing among 2 or more modules. The modules in the Digital Library of Theses are integrated or combined in a logical and systematic manner to uncover data loss and errors that occurs in interface modules.

The approach used in testing the system is the incremental approach. In this approach, the program is constructed and tested in small segments. The benefits of this testings are easier error isolation and correction, more comprehensive interface testing and a systematic test approach way be applied

Test Case 1:

After uploading a new thesis (module *submissionForm(thesis).asp*), these new theses should be able be displayed in the theses listing (module *list(year).asp*, *list(dept).asp* and *list(author).asp*). The total number of the theses must be examined.

Test Case 2:

After uploading a new thesis (module *submissionForm(thesis).asp*) or source code (module *submissionForm(object).asp* and *submissionForm(full source code).asp*), these new theses or source code should be able be searched by the search engine of the system (module *simple search(thesis).asp*, *simple search(object).asp*, *simple search(full source code)*, *advance search(thesis)*, *advance search(object)* and *advance search(full source code).asp*).

5.5.2.3 Interface Testing

The interface of the system must be user-friendly and not misleading. Errors messages must be clear and unambiguous. It is very important to ensure that the interaction of the users with the system is good.

Test Case 1:

Test whether the users understand all the functions being displayed in the menu such as "submission", "listing" and "search". For example, the "submission" function, a test being carried to examine whether a user understands clearly about the instructions given in the form and whether a user understands how to upload their theses and source code. This test was carried out by asking some students to use the system.

Test Case 2:

Use different size of computer monitor to test the system and see whether the layout of the system is acceptable.

5.5.2.4 System Testing

A system test is a test of the entire system in an attempt to exercise all processing situations. System testing involves a series of different test that is meant to fully exercise the computer-based system.

The objectives of system testing are:

- To perform a final test of all programs against the design specifications

- Ensure that the operation group has sufficient documentation and instructions to operate the system properly, process the incoming data, and distribute outgoing information from the system
- Guarantee that the end users can successfully interact with the system
- To verify that all information system components and through the computer within predicted the requirements
- Data in a timely and responsive manner

CHAPTER 6 SYSTEM EVALUATION

This chapter states all the problems faced throughout the development of the Digital Library of Theses along with their solutions. A list of the strengths and limitations of the Digital Library of Theses are also presented.

6.1 Problems Faced and Solutions

Various problems were encountered throughout the development of the Digital Library of Theses. The problems and the approaches taken to solve them are documented in the following sections.

6.1.1 Lack of Knowledge on Web-Based Programming

This is a major problem as the concept of web-based programming is very much different as compared to the traditional stand-alone application. Building web application using the three-tier client server approach adds more difficulty and complexity to the development process. Therefore the lack of exposure to web-based programming has increased my learning curve.

Surfing the Internet for information and reading up on the concept of client-server and Internet programming which included the operations of a web server, were some of the approaches taken to overcome this problem. Most of the ambiguities are resolved by reading up on relevant materials, participated in web side discussion forums and most importantly advice and guidance from course mates and the project supervisor.

6.1.2 Lack of Exposure to the Database Server

The development of the Digital Library of Theses involves the use of a database server (Microsoft SQL Server 7.0) due to the vast amount of information stored in a library. This also increases the project learning curve, as there is a need to do research on the work ways of a database server. The concept of database server is rather different that the conventional ones as there are many additional features that need to be explored such as stored procedures, trigger and task scheduling.

To overcome this problem, the Microsoft Corporation web site on SQL Server was visited to obtain more information on the database server. Besides that, reading up the relevant materials on SQL Server did help in overcome the problem.

6.2 Strengths of the System

6.2.1 Source Code and Objects Library

The Digital Library of Theses is a digital library that stores the source code in the form of entire source code, objects or components of each project. The collections of objects or components are very useful in promoting the reusability of the objects or component in the development of future projects. The full source codes stored in the Digital Library of Theses are well managed with proper description.

6.2.2 Electronic Theses Library

The Digital Library of Theses stores the theses electronically for each year. Most of the web pages in the Digital Library of Theses are generated dynamically meaning that all the information is retrieved from the database.

6.2.3 Year Listing

The Digital Library of Theses has collection of theses from year to year. All the theses are categorized according to year. The total number of theses of each year is listed. This improves the effectiveness of searching.

6.2.4 Search Engine

The Digital Library of Theses provides a search engine that enables users to search the theses, the entire source code and the components stored in the system. Users can perform a simple search or an advanced search under this search engine.

6.2.5 On-line Theses, Source Code or Components Submission

The Digital Library of Theses provides the uploading function for the users. This improves the efficiency of the system and reduces the manpower involve in managing the theses and the source code.

6.2.6 Downloadable Source Code and Components

Users can easily search the source code or the components by using the search engine and the source code or the components can be downloaded from the server.

6.2.7 Attractive and Ease of Use Graphical User Interface

The major advantage of the Digital Library of Theses over the conventional library information retrieval system is that it provides an attractive and easy to use graphical user interface. Most of the operations can be performed by point and click, hence users can navigate from page to page without much effort. Besides that the search options are also provided to allow the user to select.

6.2.8 Total User Control

The users have full control on the system, meaning they can navigate to any pages and perform any functions that are provided by the system. The users if logon, can also log off anytime they wish.

6.2.9 System Transparency

All the users need to do is just type in their query and the system will handle the searching for them without their intervention.

6.2.10 Sorting

Search results can be sorted by the users based on the field they chose. The users the can sort the search results in ascending direction.

6.2.11 Validation on Input Data

The Digital Library of Theses performs strenuous validation on user input. This further enhances the system reliability and error handling. Certain characters that are not accepted in certain cases are blocked to avoid errors from happening. Therefore users will not be bogged down by sudden errors happening at entry.

6.3 System Limitations

The following are the limitations of the Digital Library of Theses.

6.3.1 Limited Boolean Combinations For Advanced Search

Currently the advanced search provided by the search engine of the Digital Library of Theses supports only two Boolean combination and they are AND and OR.

6.3.2 Browser Limitation

Browser such as Internet Explorer that supports ActiveX and VBScript is able to run the Digital Library of Theses. Browser like Netscape Navigator will not be able to run the Digital Library of Theses.

6.3.3 No Password Encryption

The password that is sent on-line during the process of logging on to the uploading function is not encrypted. This may cause the exposure of the password to the unauthorized users.

6.3.4 No Context Sensitive Help

Digital Library of Theses does not provide context sensitive help for users. If users have any difficulty in using a specific function or module, they need to search the topic of the entire help.

6.4 Future Enhancements

Below are the suggested future enhancements to the Digital Library of Theses.

6.4.1 Enhance Boolean Searches

Currently, the Digital Library of Theses provides only two Boolean combinations of searches, AND and OR. In future, the Digital Library of Theses can be enhanced by providing more combinations searches such as NOT, NEAR and WITHIN.

6.4.2 Support Multi Languages Character Set

The Digital Library of Theses can only search for Roman characters. Thus, this system can be further enhanced to include more language character sets such as Chinese and Japanese character set.

6.4.3 Database Integrity

The system should be able to be restored automatically after the disasters like the occurrence of the power failure. Currently, the servers need to be manually started after power failure occurs.

6.4.4 Password Encryption

Password that is sent on-line should be encrypted to avoid the exposure of the users' password to the malicious users. This also ensures the confidentiality of the system information.

6.4.5 Discussion Forum

The Digital Library of Theses should provide a discussion forum for the users to enable the users to discuss their problems throughout the development of their theses.

6.4.6 Provides Interactive Help Function (demo-based or in video form)

The Digital Library of Theses can enhance its help by providing a more attractive and efficient on-line help. Presenting the help procedures in video or demonstration form by incorporating the multimedia features can help the users on how to use the system.

6.4.7 Integrating a Mail Server

Integrating the Digital Library of Theses with a mail server can further enhances the capabilities of the system. This is because a mail server helps in enabling the administrators to either reply a mail to a specific user or to put the reply in the announcement page.

CHAPTER 7 CONCLUSION

As a conclusion, the Digital Library of Theses has achieved the objectives and fulfilled the requirements of a web based digital library system as stated in the project scope and system analysis. The Digital Library of Theses allows users to upload their theses and the source code to the server. The Digital Library of Theses does provide functions such as simple search and advanced search that enable users to retrieve theses from the database and perform the download function. The use of web-based approach brings along many benefits such as the ability to access information from anywhere and at anytime of the day. It saves a lot of storage space for the theses and at the same time promotes a paperless environment. It also eases the source code storage and management. In addition, the Digital Library of Theses provides an attractive ad easy to use interfaces that greatly reduces confusion and eases retrieval work as in the traditional way of storing theses and source code.

A lot of valuable knowledge was gained especially in web-based programming throughout the development of the Digital Library of Theses. The knowledge gained includes web-based programming languages such as Active Server Pages, VBScript and JavaScript, knowledge in installing and configuring the NT Server and Web Server, the concept of the three-tier client/server architecture, understanding on the works of the web server and techniques used in implementing a database server. Other areas of knowledge gained include the area of uploading, downloading and information retrieval. Theories that have been taught and learnt such as system

analysis and design, software engineering, programming skills, data communications and algorithm analysis has been put into practical use.

The Digital Library of Theses will become more complete as the new enhancements and features such as providing support for more Boolean search that enable users to perform more combination of searches supporting more languages are incorporated in the future. It is therefore hoped that the Digital Library of Theses will serve as a foundation for future implementations of such systems in digital library system.

BIBLIOGRAPHY

[1] Drabenstott, Karen M. Analytical review of the library of the future, Washington, DC: Council Library Resources, 1994.

[2] European Initiative in Library and Information in Aerospace - EURILIA. http://www.cranfield.ac.uk./cils/library/eurilia/eurilia.htm

[3] UMI's Dissertation Explorer. http://www.umi.com/hp/Support/DExplorer/prepare/submit.htm.

[4] Common Object Request Broker Archictecture, OMG, July, 1995

[5] Microsoft Corporation, "Windows NT Server: A Platform for Computing Task", 21 October 1988 http://www.microsoft.com/ntserver/techdetails/overview/ntplatform.asp

[6] Edelstein, Herb. "Unraveling Client Server Architectures." DBMS 7, 5 (May 1994): 34(7)

[7] Schussel, George. Client/Server Past, Present, and Future [online]. Available WWW http://news.dci.com/geos/dbsejava.htm> (1995).

DCOM architecture

http://msdn.microsoft.com/library/backgrnd/html/msdn_dcomarch.htm

Distributed Component Object Model

http://whatis.techtarget.com/WhatIs_Search_Results_Exact/1,282033,,00.html?query=dom

DCOM: Overview and Architecture

http://sern.ucalgary.ca/Courses/CPSC/547/F98/Slides/Hoang/DCOM.html

component object model : a technical overview

http://msdn.microsoft.com/library/techart/msdn_comppr.htm

Security Tradeoffs: Java vs. ActiveX

http://www.cs.princeton.edu/sip/faq/java-vs-activex.html

Active X vs. Java Beans

http://xent.ics.uci.edu/FoRK-archive/summer96/0577.html

Using Active Platform to Enhance Your Web Site

http://www.gasullivan.com/boerg/00000/0005a.htm

Java or ActiveX?

http://www.squico.demon.co.uk/spiral/javaoractivexcmp.htm

Chapter 1. An Overview of the Java 2 Platform

http://www.oreilly.com/catalog/jpower/chapter/ch01.html

Java Platform

http://ei.cs.vt.edu/~wwwbtb/book/chap21/javaplatform.html

Active Platform

http://sislands.com/coin70/week10/activex.htm

Using the Component Object Model (COM)

http://www.puremeta.com/rosenmader/lar/OLE/OLEPaper.html

Matthew G. Kirshenbaum (1998). "Electronic theses and dissertations in the humanities : A directory of on-line references and resources."

http://etext.lib.virginia.edu/ETD/ETD.html

Networked Digital Library of Theses and Dissertations

http://www.ntdld.org

http://www.dlib.org/dlib/september97/theses/09fox.html

http://www.theses.org

Virginia Tech

http://etd.vt.edu

West Virginia University

http://www.wvu.edu/~theses/

University of Waterloo

http://www.and.uwaterloo.ca

http://www.lib.uwaterloo.ca

University of Toronto

http://www.fis.utoronto.ca/etd

Exon, Maggie, "Long Term Management Issues in the Preservation of Electronic Information"

http://www.nla.gov.au/niac/meetings/npo95me.html, Nov.1995