



UNIVERSITY OF MALAYA

Bachelor of Computer Science

Cross-Language Information Retrieval System
using Dictionary-based Methods

WONG JUN BILL

WEK 020288

(SESSION 2004/2005)

Perpustakaan SKTM

Final Year Thesis Project

Under supervision of

Mrs. Norisma Idris

Moderated by

Ms. Johriah Norfizlina Ismail

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 Computer Science

Abstract

This dissertation proposes a Malay-English cross-language information retrieval system which is using dictionary-based method. This system first translates a given Malay query into English query using a Malay-English dictionary, and then retrieves documents relevant to the translated English query. This CLIR system will be a useful system that allows users who want to use both Malay and English natural language query to retrieve English documents. Dictionary-based CLIR problems are discussed in this proposal and some methods have been proposed in order to address the problems. The methods are including query language identification, n-gram handling, dictionary translation, abbreviation checking and query structuring. In query translation, to derive possible translation for user query, dictionary is used and a transliteration method is performed, which generates translation for untranslatable Malay words based on n-gram based matching. This solution will improve the query translation accuracy. This system will be developed on Windows 2000 Professional and the software development tools that will be utilized are Apache web server, PHP, Internet Explorer, and will also incorporate with the MySQL Server as the database management system.

Acknowledgement

First of all, I would like to take this opportunity to express my sincere gratitude and appreciations to my respected supervisor, Pn. Norisma, for her advice and supervision throughout this preparation of this project's proposal. Without her patience and proper guidance, I would not have been able to complete the proposal for this project successfully.

I would also like to take this opportunity to express my appreciation to Ms. Johriah Norfizlina Ismail for sparing his precious time to be my moderator. He has no doubt given me invaluable suggestion for my project.

Last but not least, I would like to thank Yok Kim who is my thesis project partner. Without her commitment and co-operation, some crucial parts of the proposal cannot be done on time. I would also like to thank some other friends who have given a lot of support to me during the system development phase.

Table of Contents

Abstract.....	i
Acknowledgement	ii
Chapter 1: Introduction	1
1.1 Overview	1
1.2 Problem Definition	3
1.3 Project Objectives	4
1.4 Project Scope	5
1.4.1 Query Translation.....	5
1.4.2 Information Retrieval	5
1.5 Project Limitations.....	6
1.5.1 The Current Stage.....	9
1.5.2 Limitation of the Expected System.....	9
1.6 Project Schedule	7
1.7 Chapter Summary.....	7
Chapter 2: Literature Review	9
2.1 Analysis Studies	9
2.1.1 Case Study 1 – TITAN CLIR System.....	9
2.1.2 Case Study 2 – Clarity	12
2.2 What is CLIR?	17
2.3 Natural Language Processing Technology	19
2.4 Dictionary-Based CLIR and its Problems	21
2.4.1 Untranslatable Search Keys	22
2.4.2 Phrase and Compound Translation	23
2.4.3 Processing of Inflected Word Forms	24
2.4.4 Translation Ambiguity	24
2.5 Evaluation Metrics	25
2.6 Study on Software Development Tools	28
2.6.1 System Architecture.....	28
2.6.2 System Platforms.....	32
2.6.3 Web Servers.....	34
2.6.4 Scripting Languages	36
2.6.5 Database Management Systems.....	40
Chapter 3: Methodology.....	45
3.1 Methodology	45
3.1.1 Overview	45
3.1.2 Waterfall Model	47
3.1.3 Justification of Methodology.....	49
3.2 Information Gathering Techniques	50

Chapter 4: System Analysis.....	52
4.1 Functional Requirement.....	52
4.1.1 Query Translation System.....	53
4.1.1.1 Input Function.....	53
4.1.1.2 Output Function.....	53
4.1.2 Dictionary-Based CLIR System.....	53
4.1.2.1 Malay-English CLIR Search Engine.....	54
4.1.2.2 Documents Indexing Module.....	54
4.2 Non-Functional Requirement	54
4.2.1 Performance.....	55
4.2.2 Maintainability	55
4.2.3 Reliability.....	56
4.2.4 Portability	56
4.3 Hardware and Software Requirements	56
Chapter 5: System Design	58
5.1 System Architecture Design	58
5.2 System Structure and Components	60
5.2.1 Language Identification	60
5.2.2 Dictionary Translation	62
5.2.3 Abbreviation Checking.....	62
5.2.4 N-gram Handling	62
5.2.5 Target Language Query Structuring	63
5.3 Data Design	63
5.4 User Interface Design.....	65
5.4.1 Web Page Design.....	67
5.4.2 Screen Capture.....	68
Chapter 6: System Implementation.....	70
6.1 Development Environment.....	70
6.1.1 Hardware in the Development Environment	70
6.1.2 Software Configuration	71
6.2 Development of the System	72
6.2.1 System Development Process	72
6.2.1.1 Review the System Documentation.....	73
6.2.1.2 Design the System.....	73
6.2.1.3 Code the System	73
6.2.1.4 Test the System.....	73
6.2.1.5 Complete the System Documentation.....	74
6.2.2 Coding Principle Applied	74
6.2.2.1 Readability	74
6.2.2.2 Reusability	74
6.2.2.3 Modularity	74
6.2.2.4 Good Naming Techniques	75
6.2.2.5 Internal Documentation	75
6.2.3 Programming Method	76
6.2.3.1 Modular Programming.....	76

6.2.3.2	Structured Programming	76
6.3	Malay-English CLIR and Query Translation System	77
6.3.1	System Directory Structure.....	77
6.3.2	Functions Used	78
6.3.3	SQL Statements	80
Chapter 7:	System Testing	81
7.1	Types of Faults	81
7.1.1	Algorithms Faults.....	81
7.1.2	Throughput/Performance Faults.....	81
7.1.3	Computational/Precision Faults	82
7.2	Testing Process	82
7.2.1	Module Testing.....	83
7.2.1.1	PHP Script Debugging.....	84
7.2.2	Integrated Testing	84
7.2.3	System Testing.....	85
7.2.4	Testing Check List	86
Chapter 8:	System Evaluation	88
8.1	Evaluation on Query Translation System.....	88
8.1.1	Evaluation on Applied N-gram Handler.....	88
8.1.2	Evaluation on Query Translation System.....	90
8.1.3	Recall and Precesion	92
8.2	Difficulty and Suggested Solution.....	93
8.2.1	Problem in Finding a Better Malay-English Dictionary	94
8.2.2	Insufficient of Experience in PHP and MySQL	94
8.2.3	Time Constraint	94
8.3	System Strength.....	95
8.3.1	Acceptable for Both Malay and English query	95
8.3.2	Manageable Threshold Value	95
8.3.3	Systematic Error Handling	95
8.3.4	Easy to Use.....	95
8.3.5	Nice Interface Design	96
8.4	System Limitation	96
8.4.1	Limited Document Collection	96
8.4.2	Limited Entries of Dictionary	96
8.4.3	Slow Indexing Process	96
8.4.4	Nice Interface Design	97
8.5	Future Enhancemen.....	97
8.5.1	More Documents Added.....	97
8.5.2	Use a Good Dictionary	97
8.5.3	Better Method of Indexing Process	98
8.6	Knowledge and Experience Gained.....	98
8.7	Review on Goals	99
8.7.1	Expected Achieved	99
8.7.2	Objective Achieved.....	99

References	100
------------------	-----

Appendix A	Install Apache	
Appendix B	Install PHP	
Appendix C	Install MySQL	
Appendix D	User Manual	

Figure 2.1	CLIR system page	10
Figure 2.2	CLIR system page	11
Figure 2.3	Overview of CLIR system	13
Figure 2.4	A view of the CLIR interface	13
Figure 2.5	CLIR system architecture with respect to the number of target languages	16
Figure 2.6	CLIR system architecture with respect to supported user request	16
Figure 2.7	Definatory based CLIR system architecture	22
Figure 2.8	Translation of query	25
Figure 2.9	Set of relevant lexical and relevant topics	26
Figure 2.10	Formulas for recall	26
Figure 2.11	Formula for precision	27
Figure 2.12	Recall - Precision Curve	27
Figure 2.13	Two tier CLIR system architecture	30
Figure 2.14	Three tier CLIR system architecture	33
Figure 3.1	CLIR system architecture	43
Figure 3.2	Workflow model	44
Figure 3.3	The CLIR system architecture	45
Figure 3.4	CLIR system overview	49
Figure 3.5	Components of CLIR system	50
Figure 3.6	CLIR system page	56
Figure 3.7	CLIR search	58
Figure 3.8	Relevant topics	60
Figure 3.9	System architecture	60
Figure 3.10	CLIR system	62
Figure 3.11	CLIR system	64
Figure 3.12	CLIR system	65
Figure 3.13	CLIR system	66
Figure 3.14	CLIR system	67
Figure 3.15	CLIR system	68
Figure 3.16	CLIR system	69
Figure 3.17	CLIR system	70
Figure 3.18	CLIR system	71
Figure 3.19	CLIR system	72
Figure 3.20	CLIR system	73
Figure 3.21	CLIR system	74
Figure 3.22	CLIR system	75
Figure 3.23	CLIR system	76
Figure 3.24	CLIR system	77
Figure 3.25	CLIR system	78
Figure 3.26	CLIR system	79
Figure 3.27	CLIR system	80
Figure 3.28	CLIR system	81
Figure 3.29	CLIR system	82
Figure 3.30	CLIR system	83
Figure 3.31	CLIR system	84
Figure 3.32	CLIR system	85
Figure 3.33	CLIR system	86
Figure 3.34	CLIR system	87
Figure 3.35	CLIR system	88
Figure 3.36	CLIR system	89
Figure 3.37	CLIR system	90
Figure 3.38	CLIR system	91
Figure 3.39	CLIR system	92
Figure 3.40	CLIR system	93
Figure 3.41	CLIR system	94
Figure 3.42	CLIR system	95
Figure 3.43	CLIR system	96
Figure 3.44	CLIR system	97
Figure 3.45	CLIR system	98
Figure 3.46	CLIR system	99
Figure 3.47	CLIR system	100

List of Tables

Table 2.1	Strengths and weaknesses of Apache and IIS	36
Table 4.1	Summary of Hardware and Software Requirements	56
Table 5.1	Description of table <i>dictionary</i>	64
Table 5.2	Description of table <i>stopWord</i>	64
Table 6.1	List of functions	78
Table 7.1	Testing check list	86
Table 8.1	Evaluation of 2 N-gram handlers	89
Table 8.2	Evaluation of implemented Query Translation System	90

1.1 Overview

In these few decades, information technology is growing and faster. As an impact, people are provided a better service and they also look for an exchange information, searching information and also in a communication network among them. At this point, the issue of information retrieval (IR) system has become important. We can find a lot of examples of the IR system on the World Wide Web. The Yahoo and Google engines allow user to search information for internet. Yet, most of the existing IR systems are using monolingual interfaces where the information retrieval is only involving single language. As a result, people who use English are able to search information that only written in English and people who use Malay will search documents in only Malay. However, in some cases, we need to retrieve information that written in other languages different from the query language. From this point, cross language information retrieval (CLIR) has become the solution that overcome the problem of searching multilingual information through single language.

List of Figures

Figure 1.1	Cannot process Malay query on English IR system	3
Figure 1.2	Expected process of the Malay-English CLIR system	4
Figure 1.3	Project Schedule	7
Figure 2.1	TITAN's architecture	10
Figure 2.2	TITAN result page	11
Figure 2.3	Overview of Clarity system	13
Figure 2.4	A view of the Clarity interface	15
Figure 2.5	Clarity's response times with respect to the number of target languages	16
Figure 2.6	Clarity's response times with respect to concurrent user requests	16
Figure 2.7	Dictionary-based CLIR system architecture	22
Figure 2.8	Translation ambiguity	25
Figure 2.9	Set of relevant items and irrelevant items	26
Figure 2.10	Formula for recall	26
Figure 2.11	Formula for precision	27
Figure 2.12	Recall – Precision Curve	27
Figure 2.13	Two tier Client-server Architecture	30
Figure 2.14	Three tier Client-server Architecture	32
Figure 3.1	Generic System Development Process Model	45
Figure 3.2	Waterfall Model	47
Figure 5.1	Three-tier client/server architecture	59
Figure 5.2	CLIR system overview	60
Figure 5.3	Components of Query Translation System	61
Figure 5.4	CLIR main page	68
Figure 5.5	CLIR search page	69
Figure 5.6	Document indexing page	69
Figure 6.1	System development process	72
Figure 6.2	Comments in coding	75
Figure 6.3	System directory structure	78
Figure 7.1	Testing process	82
Figure 8.1	Recall-Precision curve for Malay query set	92
Figure 8.2	Recall-Precision curve for Malay query set	93

Chapter 1

Introduction

This chapter is emphasizes on the introductory description of this dissertation. Project overview, objectives, scope, limitation, schedule and problem definition are discussed. Besides, significance of this dissertation and its potential contributions are described.

1.1 Overview

In these few decades, information technology is growing faster and faster. As an impact, people are provided a better environment that allows them to exchange information, searching information and establish a communication network among them. At this point, the issue of the information retrieval (IR) system has become important. We can find a lot of examples of the IR system on the World-Wide-Web: the Yahoo and Google search engine allow user to search information on the Internet. Yet, most of those running IR systems are using monolingual technique where the information retrieval is only involving single language. As a result, people who use English retrieve documents that only written in English and people who use Malay will retrieve documents in only Malay. However, in some cases, we need to retrieve information that written in other languages different from the query language. From this point, cross language information retrieval (CLIR) has become the solution that can solve the problem of accessing multilingual information through single language query.

CLIR is a subfield of information retrieval dealing with retrieving information written in one or several other languages different from the language of the user's query. For instance, a user might use a query in Malay and posts it for retrieving documents which are written in English. In 1964, the first research of CLIR has been taken out for the development of International Road Research Documentation system that used a controlled-vocabulary thesaurus [3]. However, the thesaurus-based technique that used at that time did not completely solve the CLIR problem. From the year onwards, this research is receiving growing attention by computer scientists from every corner of the world.

At present, three major evaluation efforts has been established for this field: the Cross-Language Evaluation Forum (CLEF) which covering many of the European languages, the NTCIR Asian Language Evaluation which works on Chinese, Japanese, Korean and etc, the TREC Cross-Language Track which focuses on Arabic language [2]. These forums are working together and give a lot of contribution on this research. They establish a strong links, exchanging ideas and sharing results with each other and also the other similar cross-language evaluation. In order to improve the universal usability of CLIR, many issues have been sparked and studied.

1.2 Problem Definition

Across the 20th century, English language is widely used in most of the countries of the world. We cannot deny that English language is very important in our daily life and most of the information that convey to people is in English. In many fields, we can found that many of the documents are written in the language. As a result, resources which written in English are often the target retrieved result of most of the IR tasks.

In Malaysia, some courses conducted in universities are using Malay language. However, most of the resources are all in English. Students are sometimes required to do research and need to find some information for the research especially resources from the Internet. In this situation, students will face problem to search information in English because there are only monolingual IR system available (see Figure 1.1), e.g. Yahoo and Google search engine. That means they cannot use Malay query to retrieve information that written in English. In order to address this problem, a Malay-English CLIR system is needed for retrieving those information. The expected process is shown in Figure 1.2.

The research of CLIR in Malaysia is currently in the early stage. We hardly find a usable Malay-English CLIR system that available in the market. This has become a problem and also an opportunity that triggers this project. Thus, at the end of the project, we are expecting a desired outcome that can solve this problem.

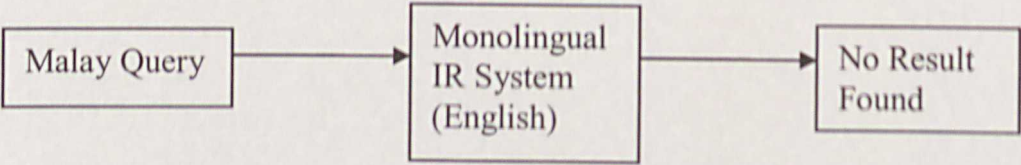


Figure 1.1: Cannot process Malay query on English IR system



Figure 1.2: Expected process of the Malay-English CLIR system

1.3 Project Objectives

The objective of this project is to study the research of dictionary-based CLIR and also the major problems of this approach. The problems are including the issues that involve dictionary translation between Malay language and English language, query matching method, so on and so forth.

Besides, the objective is to provide students or researchers a Malay-English CLIR system that can let them retrieve their desired information conveniently from the existing databases. Consequently, they will have no problem to search English information by using Malay query. Other than that, perhaps our effort on this project can contribute to the CLIR research in Malaysia.

In this dissertation, the scope will be focused on query translation module (more detail in section 1.4). For this part, a Malay-English query translation system will be developed and this system will eventually integrate with the information retrieval system.

1.4 Project Scope

Preliminary, university students and researchers will be the target users. In order to achieve this, the system will be designed as a web-based application. Target language documents are represented in text form. Those documents are collected from newspapers, WWW etc and the document topic will be focus on field of computer science. These collected information and documents will be stored in database.

This project is divided to two major parts:

- i. Query translation, and
- ii. Information retrieval

1.4.1 Query translation

In this module, a Malay-English query translation system will be developed. In order to complete this task, some researches will be conducted on how and what processes are involved in the system, e.g. normalization, n-gram handling, etc. In addition, studies and issues that related to the translation between the languages will be carried out.

1.4.2 Information retrieval

In this part, a conventional way will be followed for developing an IR algorithm. It includes a word stemmer, query matching method and etc.

1.5 Project Limitations

1.5.1 The current stage

Currently, we can find some existing usable CLIR systems on the Internet. However, those systems do not provide all language set. Malay language is one of the examples that except from the language set. This is also the reason that triggered this project.

1.5.2 Limitations of the expected system

The query translation module is an important part of a CLIR system. The module involves large scale of research and it may become a problem where we cannot develop it perfectly. This might leads to the ambiguity of the target query. Furthermore, low coverage of dictionary used will causes some problems in translation part. As a result, users might retrieve some unwanted documents and some relevant documents will not be retrieved.

The other limitation of the expected system is that all collected documents are restricted in field of computer science. This means the system is designed only for those who are looking for resources of computer science. Besides, the system only receives Malay query and then retrieves documents which are written only in English.

1.6 Project Schedule

Project Stage	Jul				Aug				Sep				Oct				Nov				Dec				Jan			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Preliminary Study and Planning																												
Literature Study																												
System Analysis																												
System Design																												
Development and Coding																												
Unit Testing																												
System Testing																												
Documentation																												
Implementation and Maintenance																												

Figure 1.3: Project Schedule

1.7 Chapter Summary

This report is mainly divided into 5 chapters with each of which has a different focus. A brief description of each chapter is as follow:

- **Chapter 1 Introduction** emphasizes on the introductory description of this dissertation. Project overview, objectives, scope, limitation, schedule and problem definition are discussed.
- **Chapter 2 Literature Review** covers topics that related to the project. It consists of two categories of research: the conceptual research such as existing systems review and the development research such as software development tools consideration.

- **Chapter 3 Methodology** reviews the techniques taken and methodology that will be implemented to solve the problems in the project in detailed.

Chapter 2

- **Chapter 4 System Analysis** discusses the requirements of developing the query translation system. The requirements discussed are including functional requirement, non-functional requirement, and hardware and software requirement.
- **Chapter 5 System Design** discusses the overall architecture of the system. It includes architectural design, data design and user interface design of the system.

2.1 Analysis Studies

2.1.1 Case Study 1 – TITAN CLIR System

TITAN (Total Information Transfer Across Nations) was developed as a web application to explore the World Wide Web in their own language. It is a cross language WWW search system that uses a bilingual dictionary to translate queries from Japanese to English and English to Japanese. Only it accepts query words in the source language of the user (English for Japanese and Japanese for English) and returns the word in their own language. The system also provides a facility to translate a list of words or their own language into English and Japanese and display them a list of United States sources (URLs) on the search documents with the documents that translated into the source language.

TITAN consists of two parts, front-end interface for the user and a central engine. The overall architecture of the system is shown in Figure 2.1.

Chapter 2

Literature Review

This chapter covers topics that related to the project. It consists of two categories of research: the conceptual research such as existing systems review and the development research such as software development tools consideration.

2.1 Analysis Studies

2.1.1 Case Study 1 – TITAN CLIR System

TITAN (Total Information Traverse AgeNt) has been developed to assist Japanese users to explore the World Wide Web in their own language. It is a cross language WWW search system that uses a bilingual dictionary to translate queries from Japanese to English and English to Japanese. Firstly, it accepts query words in the source language of the user (English and Japanese), then searches for documents that contain given words or their translation (into English and Japanese), and finally return a list of Uniform Resource Locators (URLs) of the search documents with the documents titles translated into the source language.

TITAN consists of two parts: a catalog database builder and a retrieval interface. The overall architecture of the system is shown in Figure 2.1.

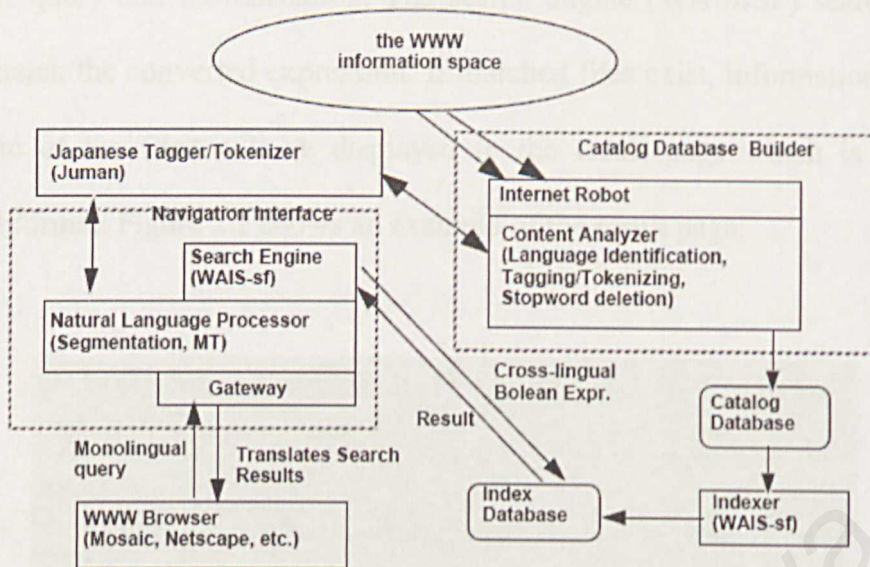


Figure 2.1: TITAN's architecture

Catalog Database Builder

This part responsible for creating a text database called a “catalog” by automatically visiting the WWW information space prior to responding user’s query. The catalog contains URLs and each of the URL is paired with a content file of the document located by the URL. The content information of a document includes words in the document, and other information that characterizes the document (e.g the language, server domain, etc). The catalog database is converted into index file using WAIS-sf indexer partially modified for Japanese language.

Retrieval Interface

The Retrieval Interface is responsible for providing an interface between the user and the catalog database. The retrieval interface interprets user queries, and then converts them into Boolean expressions (conjunctive AND or disjunctive OR) consisting of words in

statistical modules to calculate the likelihood of the text with regards to each language.

- Machine Translation (MT) module
- The MT module provides an interface in the source language of the user while performing a multi-language search on the WWW. It completes two tasks: translates the source language to the target language and translates the titles and other information of the retrieved documents.

Discussion

TITAN has been a very useful J-E bi-lingua WWW search engine which helps a lot of Japanese who do not understand English to retrieve information from the WWW which are written in English. The system provides translation of the target documents' title and other information lets user to gain more understanding of the retrieved documents. Besides, it consists of a language identifier which makes possible of many processes in the catalog builder part.

The main problems found in TITAN are those common to many other systems: the shortness of the average query and thus the lack of contextual information for disambiguation, the difficulty of recognizing and translating compound nouns [3].

2.1.2 Case Study 2 – Clarity

Clarity is an EU-funded research project with the aim to provide a cross language information retrieval system for text and audio documents. The system has been developed for English, Finnish, Swedish and Baltic languages. Similar with TITAN,

Clarity is designed as a web-based CLIR system however the system provides more language pairs. Clarity is designed as a three-layer system (shown in the diagram of Figure 2.3) with the user interface as a front-end (Interface Layer), the data sources and services on the back-end (Application Layer) and a middle layer that separates the interface from the system services and provides the communication between them (Communication Layer).

Clarity supports the following functions:

- Query translation for English, Finnish and Swedish and Baltic.
- Document retrieval and merging for multiple languages.
- Identification of query terms in each retrieved document.
- Translation of target language document titles.
- Extraction of the document's 'best passage' (text fragment of fixed length) with respect to the query terms matched in the document.

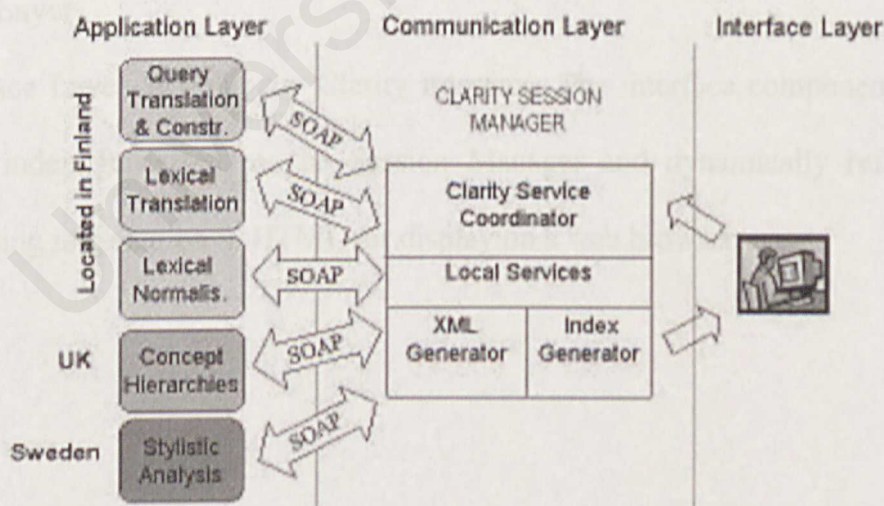


Figure 2.3: Overview of Clarity system

Application Layer

Application layer is the main part of the system which is the place CLIR located. It responsible to translate and formulate the user's query and then use the translated query to retrieve documents. The system uses the UTACLIR approach for their query translation part while documents retrieval is done by InQuary retrieval engine.

Communication Layer

The role of the Communication Layer is to act as a communicator and a data integrator between the Clarity services and the interface. The Communication Layer consists of three main parts: (1) the Web services, modeled (i.e. wrapped) around the basic services of the Application Layer, (2) the Web service clients which are used to request and receive information from the Web services, and (3) the Clarity session manager which serves as an information broker between the Clarity web services and the interface.

Interface Layer

The Interface Layer includes the Clarity interface. The interface component reads the meta-data index generated by the Session Manager and dynamically reformats the corresponding information to HTML for display on a web browser.

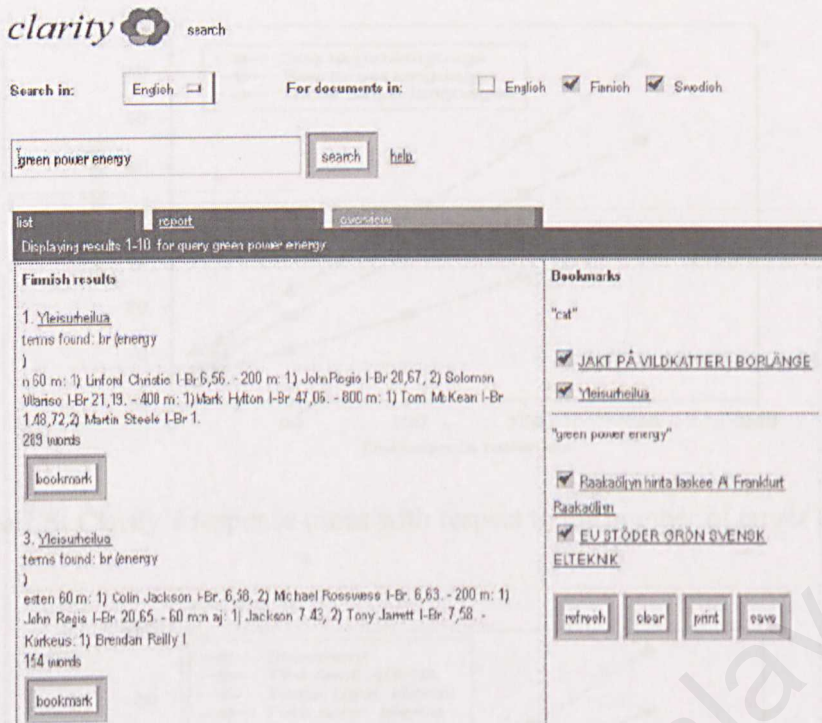


Figure 2.4: A view of the Clarity interface

Discussion

Similar to TITAN, Clarity has been designed as a Web service where several users are able to utilize the system simultaneously. In addition, Clarity is embedded the multi-lingual query translation service which able to process the source query written in the mentioned languages. However, the system has encountered a problem where features above have become constrains for the performance of the system. The number of concurrent user and selected target document language determines the respond time of the system (refer to Figure 2.5 and 2.6).

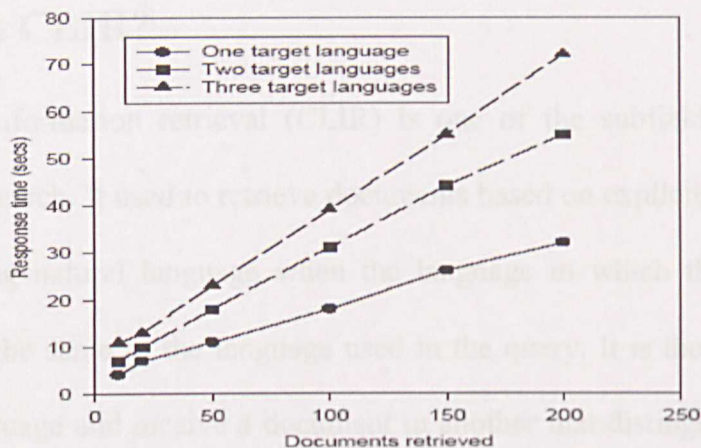


Figure 2.5: Clarity's response times with respect to the number of target languages

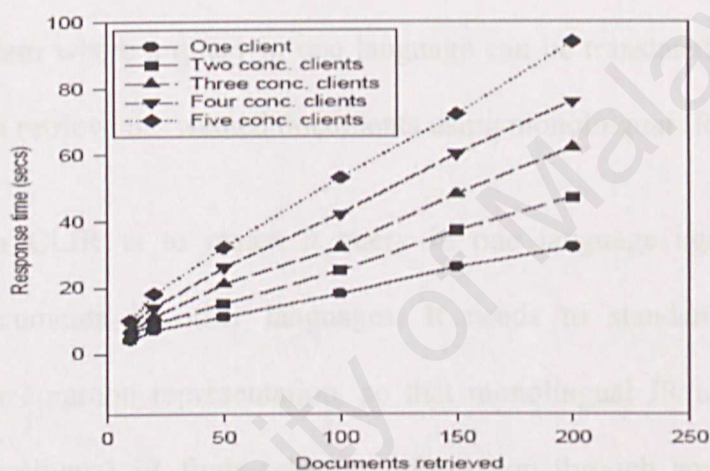


Figure 2.6: Clarity's response times with respect to concurrent user requests

2.2 What is CLIR?

Cross-language information retrieval (CLIR) is one of the subfields of Information Retrieval (IR) research. It is used to retrieve documents based on explicit queries formulated by a human using natural language when the language in which the documents are expressed is not the same as the language used in the query. It is the ability to issue a query in one language and receive a document in another that distinguishes CLIR from the conventional monolingual information retrieval. The goal of CLIR is to find the information a user needs even if it is written in a different language. This is achieved by developing a system where a query in one language can be translated to the document language and then retrieve the wanted documents using monolingual IR techniques.

The challenge in CLIR is to match a query in one language against information contained in documents in other languages. It needs to standardize queries and documents into a common representation, so that monolingual IR techniques can be applied [8]. Monolingual IR finds relevant information through some kind of word matching and weighting. In cross-language text retrieval, there are some additional problems of matching and weighting words across languages. In order to address these problems, some kinds of resources are utilized for translating the language of the query to the language of the target document, and addressing the problem of sense disambiguation. A number of approaches to this problem have been investigated by CLIR researchers, including the following:

- bilingual dictionaries
- multilingual thesauri
- machine translation

- corpus-based techniques
- parallel corpora
- conceptual interlingua
- transliteration
- translation lexicons

Each of these methods has given promising results but also has disadvantages associated with it [9]. In an attempt to overcome the limitations, the current trend system combines two or more methods to perform its translation process. For instance, it combines a bilingual dictionary together with translation-equivalent data extracted from parallel corpora. However, whatever the method chosen, the resources used to provide the means for mapping between query and collection are a major factor towards successful retrieval [9]. Although there are many problems faced by the researchers, CLIR provides a good potential for improved performance compared to that of conventional monolingual information retrieval.

Once translation of the queries is done, translated query is manipulated by a monolingual IR system for searching a collection of documents relevant to the translated query, and sorts them according to the degree of relevance.

2.3 Natural Language Processing Technology

Natural Language Processing (NLP) is one of the major fields of Artificial Intelligence. It can be seen as the capability of computers to analyze, understand, and generate languages that humans use naturally. Language processing can be divided into two tasks:

- Processing written text, using lexical, syntactic, and semantic knowledge of the language as well as any required real world information.
- Processing spoken language, using all the information needed above, plus additional knowledge about phonology as well as enough additional information to handle the further ambiguities that arise in speech.

In order to build a CLIR system, especially one that considers queries in a natural language, it is necessary to utilize NLP tools and resources. These resources are built or used as an input to the CLIR system. A dictionary based CLIR system requires resources and tools such as bilingual dictionaries, normaliser, stemmer, etc. In developing these tools and resources, it is unavoidable that there is a need to use NLP tools, resources and techniques such as morphological analysis, tagging, parsing, word sense disambiguation, semantic analysis, semantic concept representation, etc. Therefore, making NLP a crucial and important part of a CLIR system.

Several common NLP tools that used in CLIR system are:

- Stemmer
- Normaliser
- Stop words remover

Stemmer

Stemmer is the most common morphological tools for information retrieval system. It transforms words into a canonical representation by removing inflectional and derivational affixes. The resulting String is usually not a 'proper' word, but instead a stem. This is most useful for approximate matching when the actual word form is not relevant.

Since more potentially relevant documents can be retrieved, stemming can be used to improve recall in information retrieval systems. The most well known stemming algorithms are Lovins and Porter stemmers. Each of these stemmers is suitable for morphologically simple languages like English where few stem changes occur [12].

Normaliser

For compound rich languages, compound decomposition is an essential feature because of the problem with embedded search keys. If compounds are not decomposed, the non-first components are not retrievable. For instance, there are few different types of berries (blueberry, strawberry, cloudberry), all have the common tail word berry as the last component. If the compounds are split into constituents, one search key berry covers all types of berries.

The negative effect of stemming and normalising processes in information retrieval is that they may produce noise as unrelated word forms are sometimes conflated to a single form.

Stop words remover

Stop words remover uses a so-called stop word list to remove frequent non-significant words (stop words) in a document or a query string. The NLP tool has been used in monolingual IR system for removing high frequency words like articles, pronouns, prepositions conjunctions and etc.

2.4 Dictionary-based CLIR and its Problems

As discussed above, dictionary-based method is one of the approaches that used by CLIR system. This method relies on a standard bi- or multilingual dictionary that is transformed into a machine-readable form. It gives an easy applicable solution for large-scale document collections. Figure 2.7 depicts the overview system architecture of dictionary-based CLIR system.

The method makes use of the bi- or multilingual dictionary to translate each word of the source language query to the desired target language. In most of the languages, word usually consists of one or more synonyms or homonymy. Thus, after translation, translated word is represented in a set of terms. For example, the expected outcome of Malay query translation, *satu pasang kasut* (a pair of shoes) will be (one, a) (pair, install, rise) (shoes). Consequently, a problem arose where target language query will have some senses of lexical ambiguities. In order to address this problem, a particular NLP tool is needed to disambiguate the translated query.

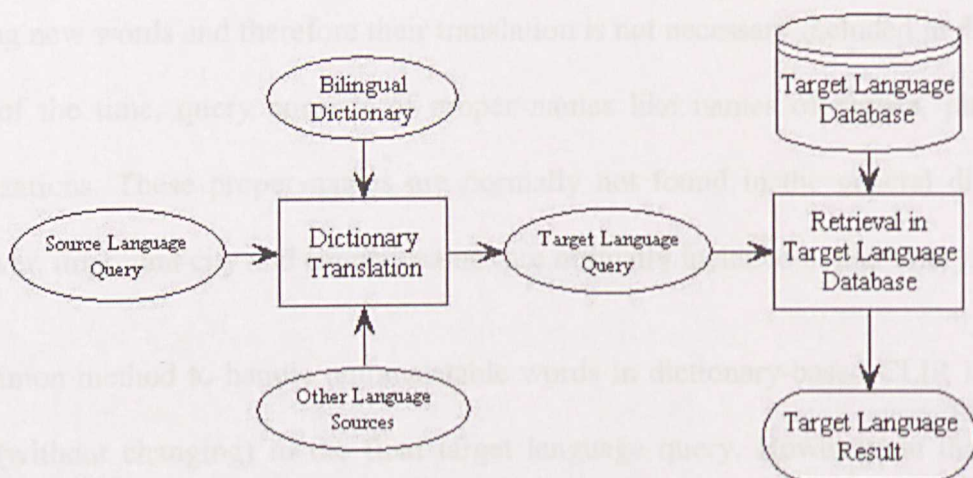


Figure 2.7: Dictionary-based CLIR system architecture

Generally, the main problems associated with dictionary-based CLIR are:

- Untranslatable search keys
- Phrase and compound translation
- Processing of inflected word forms
- Translation ambiguity

2.4.1 Untranslatable search keys

Dictionary-based CLIR uses multilingual dictionary to translate each word of the source language query to the desired target language. In some cases, word in source language query is untranslatable. One of the reasons is because of low coverage of the dictionary used. Dictionary is the most basic source of the CLIR system and its coverage determines the effectiveness of the translation. Low coverage of dictionary has not generated a big problem and it can be solved by replacing it with a better dictionary. The translation problem may also be due to compound words. Compounding is a way of

creating new words and therefore their translation is not necessary included in dictionary. Most of the time, query consists of proper names like names of person, places and organizations. These proper names are normally not found in the general dictionary. However, important city and country names are normally included in dictionary.

A common method to handle untranslatable words in dictionary-based CLIR is to pass them (without changing) to the final target language query. However, in the case of spelling variants a source language form does not match the variant form in a database index, causing loss of retrieval effectiveness [12].

2.4.2 Phrase and compound translation

Here, phrase refers to several separated words with one word sense, for instance, in Malay, *kakak ipar* means sister-in-law. In the other hand, compound word is defined as a word formed from two or more words that are written together, for example *salahguna* (misuse). Proper translation of phrases and compound words are important in dictionary-based CLIR. In this case, some techniques are needed for identifying them for further translation.

Languages in Europe are often compound-rich language. The common solution of compound word translation is first to identify the compound word, and next splitting it to its components and finally translates their components separately. Unlike European languages, Malay language consists of fewer compound words and they are usually included in dictionary.

It is more challenging to translate phrases as the process of identifying phrases is tougher. One of the solutions of this problem is using phrase-based dictionary rather than word-based dictionary. In TITAN [1], the system searches for a dictionary entry corresponding to the longest sequence of words from left to right of the query. For example, if the input string is “information retrieval system”, and if the dictionary contains “information” and “information retrieval”, then the latter entry is chosen for the leftmost part of the query. Then, it searches for the entry with “system”.

2.4.3 Processing of inflected word forms

Normally, inflected words are not included in dictionary. The commonly used method to solve inflected words translation is to remove affixes from the word form. The method is called stemming. However, the stemmed words are usually having different word sense from the word before stemming process. Thus it will increase the ambiguity of target language query.

2.4.4 Translation ambiguity

In cross language information retrieval, there are not restricted to use only one translation alternative like in machine translation. Several translation alternatives can be added to the query because words are usually consists of synonyms and homonymy. This might lead to the source and target language lexical ambiguity. As a result, irrelevant search key senses will be increased.

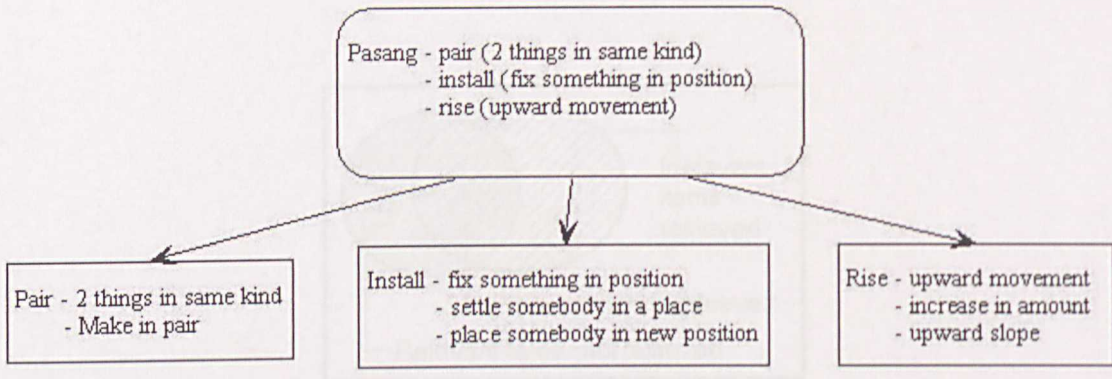


Figure 2.8: Translation ambiguity

Figure 2.8 depicts one of the translation process examples that increase ambiguity. In Malay word, *pasang* is homonymous and has 3 senses: pair, install and rise. In English word, *pair* has 2 senses: two things in same kind, make in pair; *install* has 3 senses: fix something in position, settle somebody in a place, place somebody in new position; and *rise* has 3 senses which is upward movement, increase in amount and upward slope.

The typical techniques to handle translation ambiguity in dictionary-based CLIR involve part-of-speech (POS) tagging, various corpus-based disambiguation methods, and query structuring.

2.5 Evaluation Metrics

The most often used metrics of performance that have been applied to text retrieval and information extraction are called recall and precision. These may be viewed as judging effectiveness from the application user's perspective, since they measure the extent to which the system produced all the appropriate output (recall) and only the appropriate output (precision) [11].

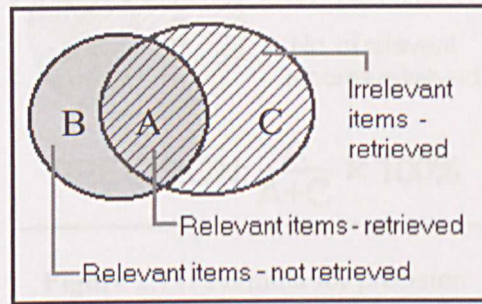


Figure 2.9: Set of relevant items and irrelevant items

Figure 2.10 shows the formula to measure recall. It is the ratio of the number of relevant documents retrieved to the total number of relevant documents in the database. It is sometimes expressed as a percentage.

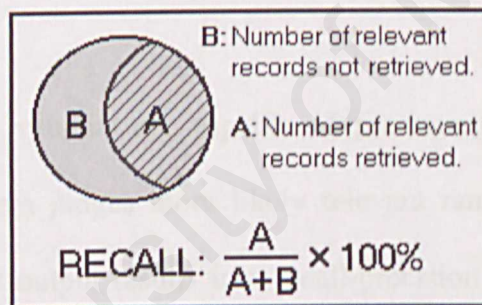


Figure 2.10: Formula for recall

In the other hand, precision is the ratio of the number of relevant documents retrieved to the total number of irrelevant and relevant documents retrieved. It is sometimes expressed as a percentage.

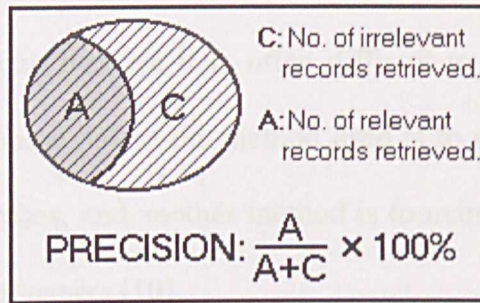


Figure 2.11: Formula for precision

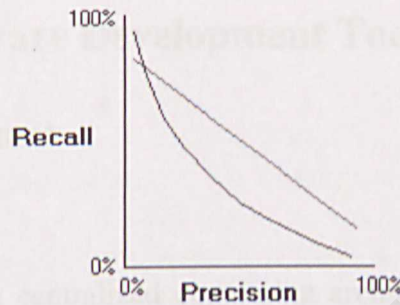


Figure 2.12: Recall – Precision Curve

Typically, text retrieval systems are capable of producing ranked results, with the documents that the system judges more likely relevant ranked at the top of the list. Evaluation of the ranked output results in a recall-precision curve, with points plotted that represent precision at various recall percentages (Figure 2.12). Recall and precision are inversely related. This means if recall is higher, then precision must be lower and vice versa.

The documents must be considered either relevant or irrelevant when calculating precision and recall. Obviously documents can exist which are 100% relevant or somewhat irrelevant. Others may be very relevant and others completely irrelevant. This problem is complicated by individual perception: what is relevant to one person may not be relevant to another [10].

Measuring recall is difficult because it is often difficult to know how many relevant documents exist in a database. Often the method used is to use all the relevant records found from different searches, and another method is to manually scan several journals to identify a set of relevant papers [10].

2.6 Study on Software Development Tools

2.6.1 System Architecture

Mainframe architecture

Mainframe architecture is a centralized computing architecture. All its intelligence is within the central host computer. Users interact with the host through a terminal that captures keystrokes and sends that information to the host. Mainframe software architectures are not tied to a hardware platform. User interaction can be done using PCs and UNIX workstations.

A limitation of mainframe software architectures is that they do not easily support graphical user interfaces or access to multiple databases from geographically dispersed sites. In the last few years, mainframes have found a new use as a server in distributed client/server architectures [6].

File sharing architecture

The original PC networks were based on file sharing architectures, where the server downloads files from the shared location to the desktop environment. The requested user job is then run in the desktop environment. File sharing architectures work if shared

usage is low, update contention is low, and the volume of data to be transferred is low. In the 1990s, PC LAN (local area network) computing changed because the capacity of the file sharing was strained as the number of online user grew (it can only satisfy about 12 users simultaneously) and graphical user interfaces (GUIs) became popular (making mainframe and terminal displays appear out of date). PCs are now being used in client/server architectures [6].

Client/server architecture

In order to address the limitations of file sharing architectures, the client/server architecture emerged. 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].

The client/server architecture can be categorized according to the number of tier implementation, starting from two-tier right up to multi-tier implementation found in today client/server environment.

Two tier architectures

With two tier client/server architectures (Figure 2.13), the user system interface is usually located in the user's desktop environment and the database management services

are usually in a server that is a more powerful machine that services many clients. Processing management is split between the user system interface environment and the database management server environment. The database management server provides stored procedures and triggers. There are a number of software vendors that provide tools to simplify development of applications for the two tier client/server architecture.

The two tier client/server architecture is a good solution for distributed computing when there are many users interacting on a LAN simultaneously. However, it does have several limitations. The performance will deteriorate when the number of users is increased to a certain number. This limitation is a result of the server maintaining a connection via "keep-alive" messages with each client, even when no work is being done. Secondly, implementation of processing management services using vendor proprietary database procedures restricts flexibility and choice of DBMS for applications. Finally, current implementations of the two tier architecture provide limited flexibility in moving program functionality from one server to another without manually regenerating procedural code [6].

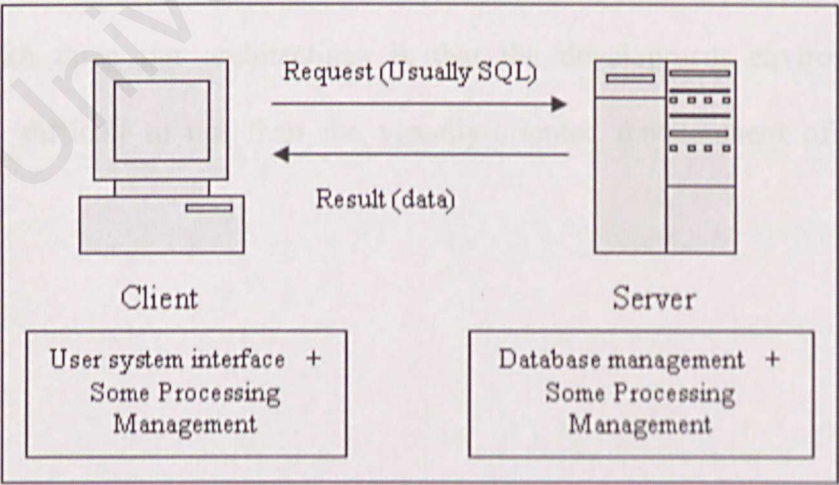


Figure 2.13: Two tier Client-server Architecture

Three tier architectures

The three tier architecture (also referred to as the multi-tier architecture) emerged to overcome the limitations of the two tier architecture. In the three tier architecture (see Figure 2.14), a middle tier was added between the user system interface client environment and the database management server environment. There are a variety of ways of implementing this middle tier, such as transaction processing monitors, message servers, or application servers. The middle tier can perform queuing, application execution, and database staging. For example, if the middle tier provides queuing, the client can deliver its request to the middle layer and disengage because the middle tier will access the data and return the answer to the client. In addition the middle layer adds scheduling and prioritization for work in progress.

The three tier client/server architecture has been shown to improve performance for groups with a large number of users and improves flexibility when compared to the two tier approach. Flexibility in partitioning can be as simple as "dragging and dropping" application code modules onto different computers in some three tier architectures.

A limitation with three tier architectures is that the development environment is reportedly more difficult to use than the visually-oriented development of two tier applications [6].

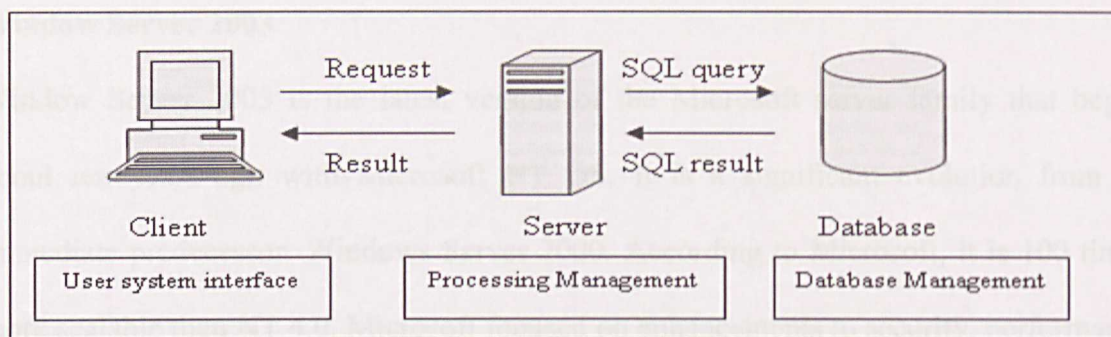


Figure 2.14: Three tier Client-server Architecture

2.6.2 System Platforms

Red Hat Linux

Red Hat Linux is a popular Linux distribution that bridges Unix and Windows worlds. It assembles open source components for the Linux operating system and related programs into a distribution package that can easily be ordered. Red Hat Linux provides many software packages, including a web server from Apache, the C language compiler from Cygnus, and the X Window system from X Consortium. It can be taken seriously as a workhouse for internet server usage, adding symmetrical multiprocessor (SMP) support, improved NFS performance, and enhanced RAID support.

In addition, interrupts are dynamically routed to all CPUs, and interrupt handlers are fully threads. Red Hat Linux, with a host of performance enhancement that will benefit web sites and internet sites of all size, is a stable and high performance operating system for internet usage.

Window Server 2003

Window Server 2003 is the latest version of the Microsoft server family that began about ten years ago with Microsoft NT OS. It is a significant evolution from its immediate predecessor, Windows Server 2000. According to Microsoft, it is 100 times more scalable than NT 4.0. Microsoft focused on enhancements to security, performance, reliability and Active Directory.

Microsoft characterizes Window Server 2003 as a multipurpose operating system capable of handling a diverse set of server roles, depending on user needs, in either a centralized or distributed fashion. Some of these server roles include:

- File and print server.
- Web server and Web application services.
- Mail server.
- Terminal server.
- Remote access and virtual private network (VPN) server.
- Directory services, Domain Name System (DNS), Dynamic Host Configuration Protocol (DHCP) server, and Windows Internet Naming Service (WINS).
- Streaming media server.

UNIX

UNIX is an operating system that originated at Bell Labs as an interactive time-sharing system. It is a layer between the hardware and the applications that runs on the computer and includes traditional operating system components.

UNIX is an ideal platform for running mail servers, networked file systems and many more at a very low cost. UNIX users normally share processing time on a central

computer or cluster or computers. This is a very cheap solution for a large number of users.

2.6.3 Web Servers

Apache

Apache is currently the leading UNIX web server. This particular web server is quick at handling requests and responses. It is available as open source and is definitely free of charge. Apache provides full source code and this encourages user feedback through new ideas, bug reports and patches.

The Apache web server is a powerful, flexible, HTTP 1.1 compliant server that implements the latest protocols, including HTTP 1.1. It is cross platforms where it runs on Linux, Windows NT/9x, OS/2 and most versions of UNIX, etc. Moreover, Apache is highly configurable and extensible with third party modules and can be customized by writing 'modules' using Apache module API. The features that implemented by Apache including DBM database for authentication, customized responses to errors and problems, multiple DirectoryIndex directives, unlimited flexible URL rewriting and aliasing, content negotiation, virtual hosts and configurable reliable piped logs.

Apache is believed to be substantially faster, more stable and more feature full than many other web server. Nowadays, Apache is runs on sites that get millions of hits per day and they have no performance difficulties.

Internet Information Services (IIS)

Internet Information Server (IIS) is the largest web servers available from Microsoft. It is a group of internet server which include a web or Hypertext Transfer Protocol server and a File Transfer Protocol server with additional capabilities for Microsoft's Windows NT and Windows 2000 server operating system. In other word, IIS is a file and web application server that can be used on local area network (LAN), wide area network (WAN) or the Internet.

IIS provides some main services as the following:

- www : world wide web service
- FTP : File Transfer Protocol service
- SMTP : Simple Mail Transfer Protocol service
- NNTP : Network News Transport Protocol service

The benefit of IIS is that it provides a high speed, secure platform for publishing information on internal networks or Internet. The server is specifically designed to provide the kind of performance that is necessary for handling an increased number of web users. IIS is easy to install. It works closely with the Microsoft transaction server to access databases and provides control at the transaction level.

Apache VS IIS

Table 2.1: Strengths and weaknesses of Apache and IIS

	Apache	IIS
Strengths	Freeware, good performance and security, working on most of the OS, reliability and extensibility, support for HTTP 1.1 Protocol, Quick technical support via Usenet newsgroup.	Free download, easy to configure, well integrated server administration tools, HTTP 1.1 support, Virtual Server Support, excellent combination with Windows NT.
Weaknesses	NT version is in its infancy, lack of graphical administration tools for configuration and administration tasks.	No Unix version and do not support by most OS, NNTP does not support USENET feeds; SMTP does not support POP3 mailboxes.

2.6.4 Scripting Languages

PHP

PHP is stand for PHP: Hypertext Preprocessor (recursive acronym) and it is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML. It is free for download and works very well on all of the UNIX-like, Linux and Windows platforms. Since PHP pages work well on all of those platforms, a particular strength is that it can be used to develop websites on a desktop system and then deployed on industrial-strength and secure servers such as those found commonly only running UNIX or Linux.

PHP is usually combined with Linux, Apache and MySQL to become a leading online application platform which is called LAMP (acronym of the four technologies) platform. Since none of Linux, Apache, MySQL or PHP requires any form of commercial license and all are battle-proven on high volume websites, these are very attractive options for those who come from the world of highly-priced proprietary packages.

PHP has a thriving developer community, numerous contributors, a wide range of features and is used in thousands of websites around the world. The range of back-end databases supported by PHP meets all but the most unusual need. PHP has proved itself easy to learn and tends to be particularly popular with those who have to deliver results rapidly and who aren't necessarily committed to taking advanced courses in programming languages.

The PHP disadvantage is its weak abstraction for databases (each database back-end is accessed through a different interface) and underdeveloped library mechanism (lacking Perl's true 'module' concept). Whereas, its benefits are its ease of deployment, ease of use and huge body of support in the industry.

Perl

Perl is one of the longest running and most successful free open-source projects. Embracing both a programming language and a whole philosophy of usage and design, Perl has quietly and without much hype, provided a better case-study of extensible and reusable software design than almost anything that has gone before it. It is very hard to find a computer system that hasn't some form of support for Perl available. Those who

like to think of themselves as serious programmers, or programming language specialists, tend to speak in very high terms of the language.

Perl is one of the best choices in the long run, especially where a considerable volume of software development and site maintenance is required, i.e. for large, complex sites subject to many and radical changes. While the disadvantage of Perl is that it's hard for non-programmers who want to complete simple tasks with active websites, requiring rather more learning before starting than say ASP or PHP.

ASP/VBscript

ASP (Active Server Pages) is a rounded proprietary product from Microsoft. Although ASP is really a framework into which various languages can plug, most people consider it implies using VB Script language. Regrettably, it is only properly supported on Microsoft's IIS platform.

ASP and VB Script are closed-source, so any bugs or security advisories that are discovered will be fixed by the Microsoft (usually slow). In the Perl or PHP worlds, it would have been quicker find the cause of the bug, fix it, submit it back to the maintainers and then carry on.

VB Script uses Active Data Objects and the ODBC interface to provide good database independence. It is an adequate programming language, but is liked even less than PHP. Simple websites are easy to build and there is a range of supporting tools to help beginners. However, it is not cross-platform thus will impact the portability and other features of the system built.

JSP

JavaServer Pages (JSP) technology allows web developers and designers to rapidly develop and easily maintain, information-rich, dynamic web pages that leverage existing business systems. As part of the Java family, JSP technology enables rapid development of web-based applications that are platform independent. JavaServer Pages technology separates the user interface from content generation enabling designers to change the overall page layout without altering the underlying dynamic content.

JavaServer Pages technology uses XML-like tags and scriptlets written in the Java programming language to encapsulate the logic that generates the content for the page. Additionally, the application logic can reside in server-based resources (such as JavaBeans component architecture) that the page accesses with these tags and scriptlets. Any and all formatting (HTML or XML) tags are passed directly back to the response page. By separating the page logic from its design and display and supporting a reusable component-based design, JSP technology makes it faster and easier than ever to build web-based applications.

JSP technology builds on the strength of the Java technology family and the multi vendor Java community. It extends the core capabilities of the Java platform to create powerful, flexible, and easy-to-maintain dynamic web pages.

JavaScript

JavaScript is an interpreted programming or script language originated at Netscape. JavaScript is interpreted at a higher level, is easier to learn than Java, but lacks some of the portability of Java and the speed of byte code. It is a cross-platform scripting language, having the properties of simple, interpreted and object-oriented.

JavaScript is a client-side scripting language which means it is running on users' web browser. The scripting language is usually used to perform some effects on a website (i.e. automatically change a formatted date on a web page, cause a link-to page to appear in a popup window and cause text or a graphic image to change during a mouse rollover).

2.6.5 Database Management Systems

MySQL

MySQL is an open source relational database management system (RDBMS), means that it is free for download from Internet and users are allowed to use and modify the software. MySQL database management system contains a large amount of functionality and power, and is noted mainly for its speed, reliability and flexibility.

MySQL provides application programs interfaces (APIs) for C, C++, Eiffel, Java, Perl, PHP, Python and Tcl. It also allows for many column types and offers full operator and function support in the SELECT and WHERE parts of queries. A complex set of databases and tables can be developed just using a simple set of command for inserting, retrieving, deleting and updating data. MySQL is cross-platform and it is available on a variety of UNIX platforms, Linux, Windows NT, Windows 95/98 and Windows 2000.

Below are the key facts of MySQL:

- MySQL can be accessed and manipulated from many popular programming languages.
- MySQL is completely optimized for both Unix and Win32 platform. It uses in-memory hash tables, thread-based memory allocation and kernel threads that are capable of utilizing multiple processors, and highly optimized individual pre-compiled class libraries.
- MySQL contains built-in support for common field type.
- MySQL supports a subset of advanced querying and grouping functions
- MySQL allows per-server password allocation
- MySQL supports a variety of connection methods, for example unit sockets and TCP/IP sockets.

Microsoft SQL Server 2000

Microsoft SQL Server is Microsoft's leading Windows database and data-warehousing package. It is the most robust database for the windows family, the Relational Database Management System (DBMS) of choice for a wide range of corporate customers and Independent Software Vendors (ISVs) building business application.

Microsoft SQL Server is a traditional database management system and is designed to understand Structure Query Language (SQL). It can manage a large amount of data in a multi-user distributed client-server environment. SQL Server is highly integrated with

the Windows NT operating system to make use of native operating system thread that can results in better performance and stability of SQL Server.

Microsoft SQL Server maintains referential integrity and security and ensures that operation can be recovered in the event of numerous types of failure. SQL Server can control the access for the type of info that can be retrieved by the user.

Below are the key facts of Microsoft SQL Server 2000:

- SQL Server 2000 is highly scalable, with a single product, from mobiles and single-user PCs through to clustered SMP and federated systems.
- SQL Server 2000 takes advantage of a number of the features of Windows 2000, to which it is closely tied.
- SQL Server 2000 does not include the extended relational and object oriented features that are now common amongst its rivals and it does not compete directly in that space. However, extended text searching facilities are included.
- The bundling of Analysis Services, Meta Data Services and ETL (Extract, Transform and Load) tools within SQL Server 2000 is a very powerful offering in so far as the Business Intelligence market is concerned.
- The Meta Data Services facility is only used within the context of SQL Server Data Warehousing and is primarily for third-party use. It is not used within a relational context for the data dictionary.

Oracle is the world's leading vendor of database software and has the distinction of being the first company to create and sell a commercial RDBMS that used SQL. Oracle database is the most scalable and full-featured database available. Whether driving the web site, packaged applications, data warehouses or OLTP applications, Oracle Database is a foundation technology for any professional computing environment.

Oracle 9i now includes all the major features (whether bundled or optional) that are available in the products of its major rivals. The only exceptions are that it does not support federated databases and that it does not include repository capabilities for supporting development environments. On the other hand Oracle 9i includes significant parts of a content management solution, which will be useful for those that wish to build their own solution in this area rather than buy a packaged product.

Below are the key facts of Oracle 9i:

- Oracle 9i now includes comparable facilities to both its major rivals in the business intelligence arena. That is, it now includes OLAP, ETL and data mining tools within the product offering.
- It consists of advanced partitioning options which are the new feature of the database management system.
- XML may be parsed prior to storage. This will reduce query overheads which would otherwise require real-time parsing as a part of the query process.
- Real Application Clustering is particularly powerful. However, it competes with the federated database capabilities offered by its major rivals.
- Oracle has provides content management support directly from the database.

- Oracle 9i consists of Internet File System feature. The true significance of this was, arguably, overlooked when it was introduced in 8i. Now it is clear that it enables a transparent consolidation capability in a particularly attractive manner.

This chapter reviews the techniques taken and methodology that will be implemented to solve the problems in the project as detailed.

3.1 Methodology

3.1.1 Overview

Methodology is an early phase in development work. Choosing and adapt a methodology for a software development project is very important and careful. Improper choice of methodology for a project can lead to failure. There are two major reasons of causing an IT project failure. The first reason is that too many design flaws are being discovered during development. It is difficult, expensive, and sometimes impossible to remedy. The second problem is that the scope of many projects seemed to expand continuously as the project proceeded.



Figure 3.1: General System Development Process

Chapter 3

Methodology

This chapter reviews the techniques taken and methodology that will be implemented to solve the problems in the project in detailed.

3.1 Methodology

3.1.1 Overview

Methodology is an early phase in development system. Choosing and adapt a methodology for a software development project is very important and crucial. Improper choice of methodology for a project can lead to a failure. There are two major reasons of causing an IT project failure. The first problem is that too many design flaws are being discovered during development where it is difficult, expensive, and sometimes impossible to remedy. The second problem is that the scope of many projects seemed to expand uncontrollably as the project proceeded.

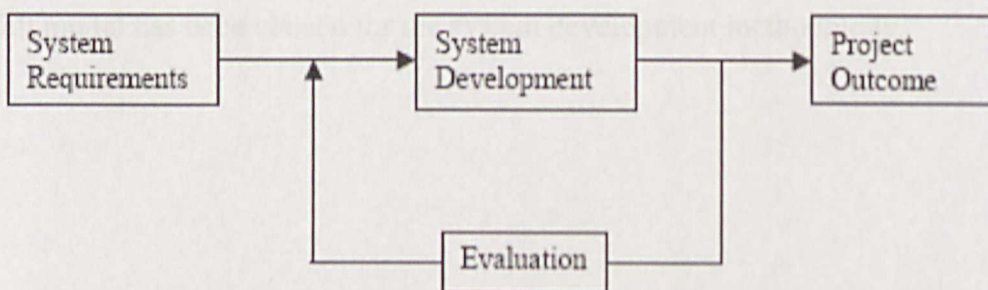


Figure 3.1: Generic System Development Process Model

Every system development process model includes system requirements such as users, constraints, and resources as inputs and a fully developed system or software as the outputs. There are many popular software process models such as:

- Waterfall Model.
- Waterfall Model with Prototyping.
- Incremental and Iterative Model.
- Prototyping.
- Spiral Model.
- V Model.
- Extreme Programming (XP) Model.
- Phased Development Model.
- Operational Specification Model.
- Transformational Model.

As discussed in chapter 1, this project is divided into two major modules which are dictionary translation module and information retrieval module. These two modules are handled and documented by two different dissertations. In this dissertation, the focus will be on query translation module. In order to develop the query translation system, the waterfall model has been chosen for the system development methodology.

3.1.2 Waterfall model

This methodology is so called waterfall because each phase flows naturally into the next phase like water over a series of fall. Each phase in the waterfall method should be completed before moving on to the next. It is attributed with providing the theoretical basis for other process models, as it most closely resembles a “generic” model for software development.

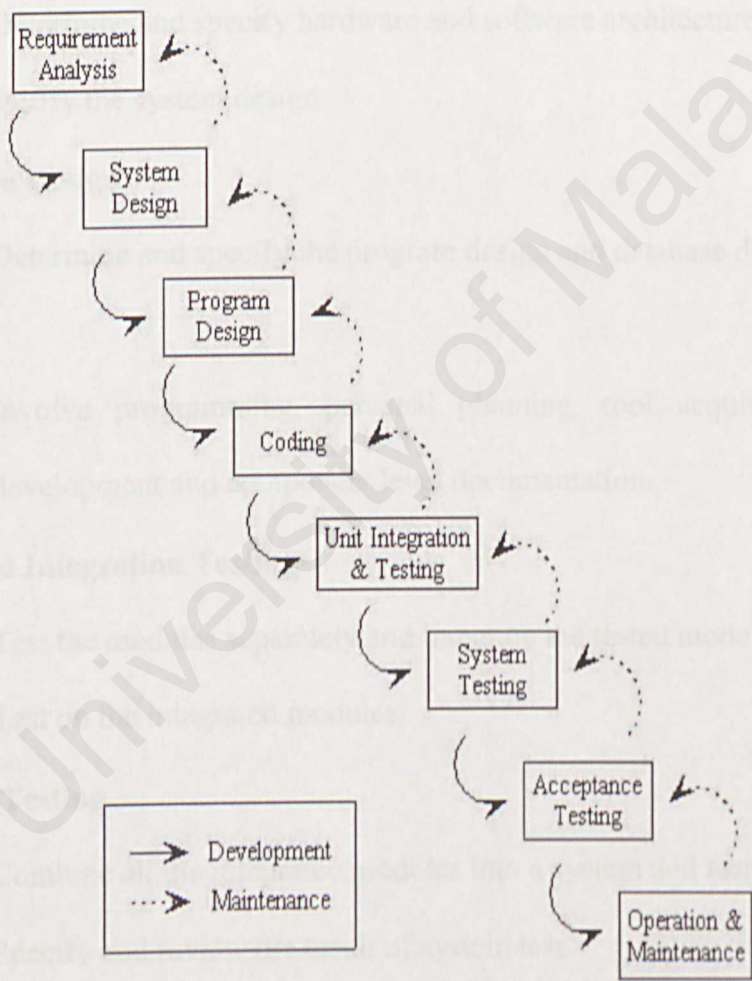


Figure 3.2: Waterfall Model

The 8 stages that illustrated in figure 3.2 are:

- **Requirements Analysis**

- Understand and determine the user's needs by having brainstorming, eliciting and analyzing system requirements.

- **System Design**

- Outlining system functionalities by having feasibility studies and case studies on current systems.
- Determine and specify hardware and software architecture.
- Verify the system design.

- **Program Design**

- Determine and specify the program design and database design.

- **Coding**

- Involve programming, personal planning, tool acquisition, database development and component level documentation.

- **Unit and Integration Testing**

- Test the modules separately and integrate the tested modules.
- Test on the integrated modules.

- **System Testing**

- Combine all the integrated modules into a system and test on the system.
- Specify and review the result of system test.
- Evaluate the system to meet the requirements.

- **Acceptance Testing**

- Testing on system is completed. The system is delivered.

- **Operation and Maintenance**

- Control and maintain the system.

The validation during system testing is to ensure that the system has implemented all the requirements, so that each system function can be tracked back to a particular requirement in the specification. As for the verification, it ensures that each function works correctly.

3.1.3 Justification of Methodology

The modified waterfall model was chosen for the following reasons:

- The development methodology is simple and easy to understand. It helps the developers to lay out what they need to do easily. Therefore in the process, they no need to burden themselves with the upcoming stage. In addition, one can has a better understanding and clearer guideline on what he or she should do during the development process. Moreover, it is easier to associate and identify each milestone with its deliverer.
- This development methodology will ensure that the developer building the right system according to the specification and verification checks the quality of the implementation. It also enables developers to develop more accurate system according to the user's discretion. This would help the developers to learn about the system and gain better understanding of the entire system.

- The arrows between adjacent stages are bi-directional. Bi-directional means that there is feedback between stages. If a problem is discovered in one stage, developers can return to the previous stages so that suitable corrective action can be taken. There is a cascading effect where developers can go back further and further up the waterfall until the problem can be corrected properly
- As the methodology is simple and easy to understand, it will be easier to present or explain to the users, especially those who are not familiar with software development.

3.2 Information Gathering Techniques

Internet

Internet searching has been the most important technique that I used in this project. A lot of researches have been done through Internet to gather useful information about this project. Without Internet, we are extremely difficult to study on the existing systems which are always a web-based application. Besides, some ideas from the existing website and its design, and some useful development tools have been recognized in order to develop this project.

Brainstorm

During the requirements elicitation, discussions with the supervisor have been carried out in order to discuss about the requirements of the system. Abundance of ideas about the design and requirements of the system have been figured out.

Observation and Informal Interview

An interview will always be useful and more interactive way to gather information. Throughout the system development phases, some informal interviews has been carried out with Malay friends and student from Akademik Pengajian Melayu for gather information like Malay language word form and other knowledge in the language etc. Besides, study and observing the existing systems are helping us a lot in designing and develop the system.

Reading Material

A lot of published literatures have been read in order to gather information of the users' need, system development needs and technical issues of the proposed system. All these can be categorized into printed material such as book and journal and non-printed material such as electronic document. Ideas are managed to get from books, magazines and journal through reading. These ideas can be implemented in the proposed system.

Chapter 4

System Analysis

In this section, the requirements of developing the query translation system will be discussed. The requirements discussed are including functional requirement, non-functional requirement, and hardware and software requirement.

4.1 Functional Requirement

In this section, functional requirement of both query translation system and dictionary-based CLIR system will be discussed.

4.1.1 Query Translation System

The query translation system will process the source language query which is the Malay query and it will be eventually translated to the target language query with a structured format. That means the system will start perform at the beginning part of the Malay-English CLIR system and end after it has produced the structured English query.

4.1.1.1 Input Function

The input of the system is a Malay query. System expects a Malay sentence or words from the users. The system do not process a Boolean query where AND or other Boolean word operator will be treated as a query content and non-characters i.e. & or other Boolean operators will be removed before proceeding to the next process.

4.1.1.2 Output Function

The output of the system is a structured English query where the query will be used for retrieving information from the English documents database. Unlike Machine Translation, the dictionary-based translation is not restricted to produce only one translation alternative. Instead, several translation alternatives can be added to the query because words are usually consists of synonyms and homonymy. So there are necessary to structure the target language query such that synonyms and other lexical part can be recognized by the information retrieval system.

4.1.2 Dictionary-based CLIR System

The dictionary-based CLIR project has a very simple requirement: to provide users a Malay-English CLIR search engine, means that users are allowed to retrieve English documents in the databases by using their Malay query. However, in order to make the system more usable and manageable, some additional sub-modules have been added.

4.1.2.1 Malay-English CLIR Search Engine

After the query translation system is completed, it will integrate with a monolingual information retrieval system to perform Malay-English CLIR search task. The features and requirements of this search engine are:

- Provide user an interface to enter their query for searching their wanted documents.
- The system should lists out all the document hyperlinks (which will link user to retrieved documents) in the result page according to the weight.

4.1.2.2 Documents Indexing Module

The requirements of documents indexing module is that administrator can index a full text document to an index form and then store in the database.

4.2 Non-Functional Requirement

Non-functional requirements describe the constraints and restriction of the system that limit the choices for constructing a solution to the problem. These constraints narrow the selection of language, platform, or implementation techniques or tools. The following non-functional requirements have been considered for the query translation system.

4.2.1 Performance

The query translation process will always look up to the bilingual dictionary. Therefore, the response time of looking up dictionary is greater contribute to the overall respond time. In order to solve this problem, an effective dictionary look up method must be used.

The bandwidth connection between user PC and server, and also performance of both user PC and server determine the system response time. Since many users will access to the system (server) simultaneously, hardware requirement of the server has become an issue. The server should have a better quality of memory and processor in order to increase the performance of the system.

4.2.2 Maintainability

System maintenance always requires more effort and time if the system is not well planned and designed at the beginning. System maintenance is a must for this system, just like any other systems, as it allows certain changes or modifications to be made over the system. For example, some better methods may be added into the query translation process. Furthermore, natural language is always changing over the time and thus many processes involved will be changed as well. Therefore, source codes and all the development phases must be documented. Source codes will be arranged and given comments according to the standard.

4.2.3 Reliability

The reliability of the query translation is determined by the bilingual dictionary and also the other language sources. The project will not expect a 100% reliability of the translation. However the system is doing its best to produce a promising reliability.

4.2.4 Portability

The system source code will be written in PHP scripting language. PHP engine is cross-platform and thus the system is portable as well. However, the recommended platforms are either Linux or Window server.

4.3 Hardware and Software Requirement

The system will be implemented by using a server. The software and hardware requirement of the server should be specified in order to ensure the implementation of the system. Table 4.1 shows the summary of hardware and software requirements that have been considered for this project.

Table 4.1: Summary of Hardware and Software Requirements

	Development Environment	Runtime Environment
Hardware Requirements	<ul style="list-style-type: none">▪ Pentium 4 2.4 GHz processor▪ 512 MB DDR SDRAM▪ 10 GB Hard Disk▪ Standard input and output devices	<ul style="list-style-type: none">▪ Pentium 4 3.0 GHz processor▪ 512 MB DDR SDRAM and above▪ 10 GB Hard Disk and above▪ Standard input and output devices.
Software Requirements	<ul style="list-style-type: none">▪ Windows 2000 Professional▪ Apache HTTP Server 4.0.49▪ MySQL Database Server	<ul style="list-style-type: none">▪ Windows 2000 Professional▪ Apache HTTP Server 4.0.49▪ MySQL Database Server 4.0.20

	4.0.20 <ul style="list-style-type: none"> ▪ PHP Engine 4.3.8 ▪ Internet Explorer 6.0 ▪ MySQL Control Center ▪ Macromedia DreamViewer ▪ Zend Studio ▪ Note Pad 	<ul style="list-style-type: none"> ▪ PHP Engine 4.3.8
--	---	--

For client side, a web browser is needed for accessing to the Apache HTTP server. There have no specification for client hardware requirement. However, for a better performance of the client PC, a Pentium 2 PC is recommended.

8.1 System Architecture Design

The CLR system uses three-tier client-server architecture as follows:

- Provides for more flexible resource allocation and more functional services
- More highly portable and can be dynamically added and deleted the work of the project change. Network can be reduced by having the program independent of the hardware to be used by the server.
- Having separate client and server allows for the parallel development of the two parts of the system.
- Modular design architecture can be used by other applications. Besides that, modular development offers several advantages. The program is tested and debugged in small units when writing these applications.

Chapter 5

System Design

In this phase, the overall architecture of the system will be discussed. It includes architectural design, data design and user interface design of the system.

5.1 System Architecture Design

The CLIR system uses three-tier client/server architecture due to few reasons: -

- Provides for more flexible resource allocation. Middle-tier functionality servers are highly portable and can be dynamically allocated and shifted as the needs of the project changes. Network traffic may be reduced by having functionality servers strip data to the precise structure required before distributing it to individual clients at the LAN level.
- Having separate software entities allows for the parallel development of individual tiers.
- Modularly designed middle-tier code modules can be reused by several applications. Reusable logic reduces subsequent development efforts, minimizes the maintenance workload, and decreases migration costs when switching client application.

The CLIR web site is designed as web-based client/server architecture. It has been divided into three-tier: applications layer, functionality tier and database repository. The first tier is the application layer that consists of all the necessary application. A web browser is the main application component that appears to the user (See figure 5.1).

The middle tier is functionality tier. The Hypertext Transfer Protocol (HTTP) plays an important role in the web pages transfer for the communications between the first tier and the middle tier. The functionality tier consists of the components that will be created to support the CLIR web site such as password verification, CLIR search engine etc. The scripting language for the system is PHP. PHP will process the request from the client and convert the result in HTML format. The user does not interact with the software in this layer. However, the validation of data is performed at this level and it is applied to the data that is either coming from or going to the web server.

The third-tier is the data repository. This layer is all about the storing of data in MySQL Server. The data like stop word list, dictionary entries and documents are stored in the databases server.

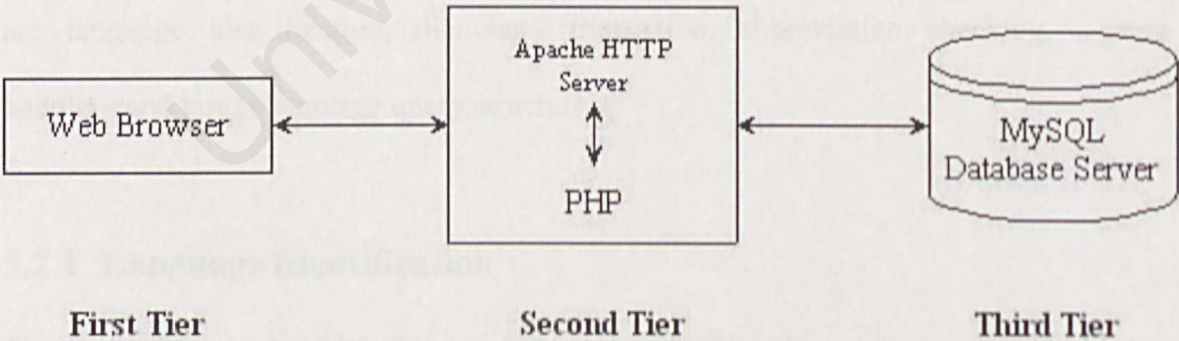


Figure 5.1: Three-tier client/server architecture

5.2 System Structure and Components

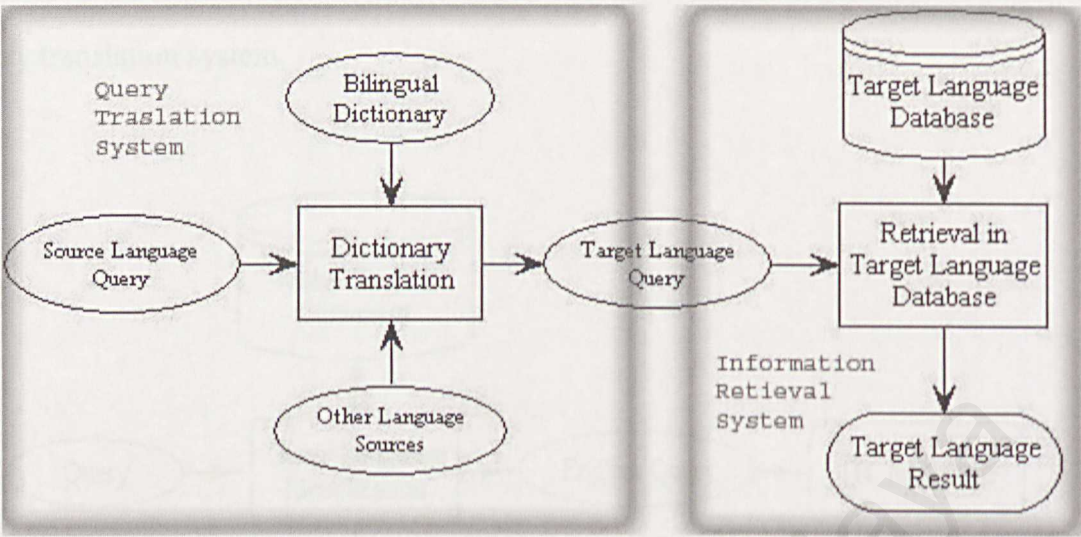


Figure 5.2: CLIR system overview

Figure 5.2 illustrates the overall system architecture of the dictionary-based CLIR system. In this section, the focus is on query translation part. The part will handle the source language query until the structured target language is produced.

As shown in Figure 5.3, the input is a Malay query; processed word by word and the output is a structured English query. The components or processes shown in the figure are language identification, dictionary translation, abbreviation checking, n-gram handling and target language query structuring.

5.2.1 Language Identification

The first process would be the query language identification. After user sends his/her query, language identifier checks the query language and determines whether it is either English or Malay. If the query language is English (target language), query translator

will no process it in further but sends it to information retrieval module directly. In the other hand, if Malay language is identified, the query will be send to other processes in query translation system.

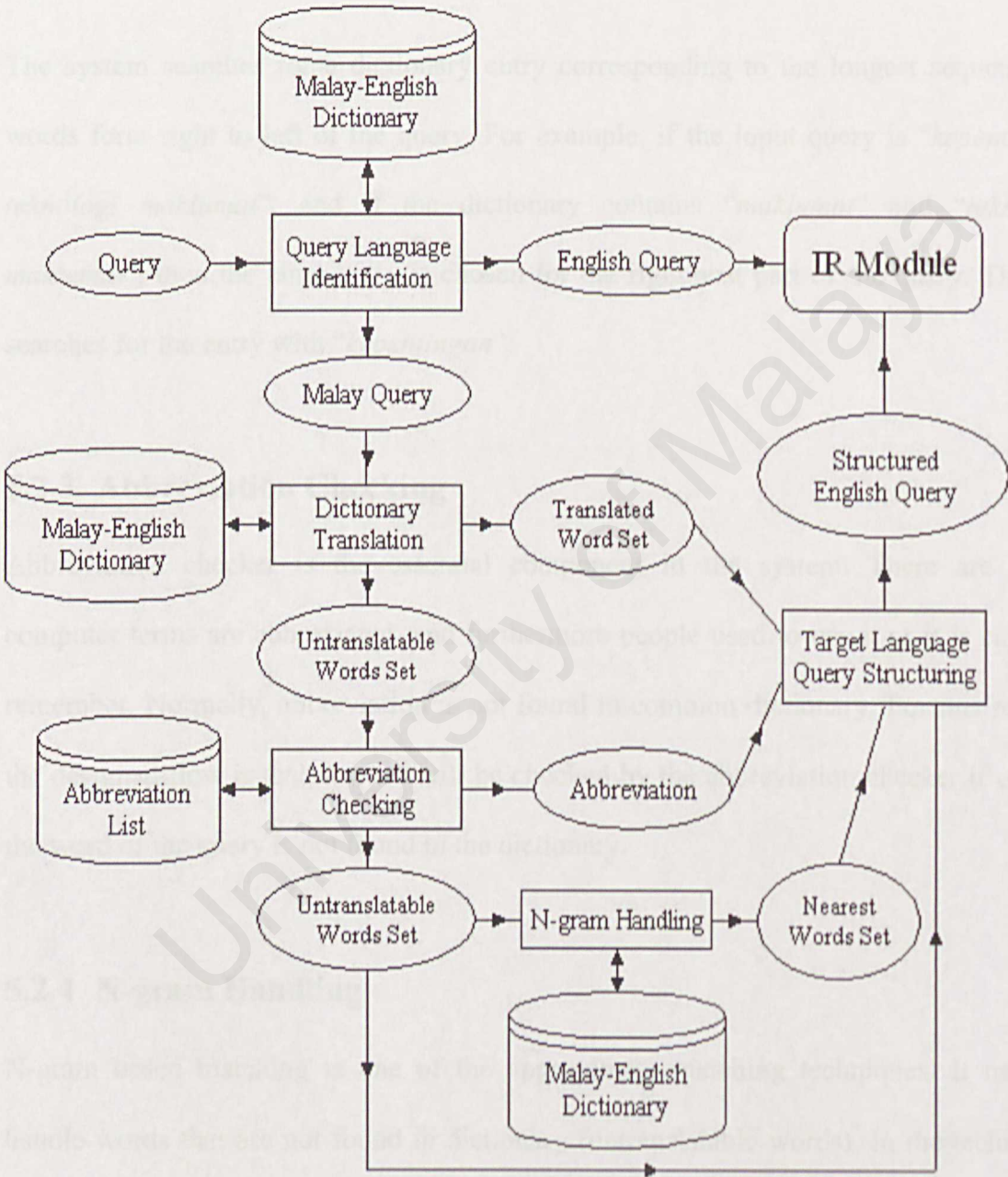


Figure 5.3: Components of Query Translation System

5.2.2 Dictionary Translation

The Malay-English dictionary is the source used for the dictionary translation. It is a phrase-based dictionary and its coverage includes compound words (e.g. *salahguna*), phrase (e.g. *kakak ipar*) and inflection words (e.g. *menggunakan*).

The system searches for a dictionary entry corresponding to the longest sequence of words from right to left of the query. For example, if the input query is “*kepentingan teknologi maklumat*”, and if the dictionary contains “*maklumat*” and “*teknologi maklumat*”, then the latter entry is chosen for the rightmost part of the query. Then, it searches for the entry with “*kepentingan*”.

5.2.3 Abbreviation Checking

Abbreviation checker is the essential component in the system. There are many computer terms are abbreviated, and furthermore people used to use it as it is easy-to-remember. Normally, abbreviation is not found in common dictionary. For this reason, the designed flow is that, a word will be checked by the abbreviation checker if only if the word of the query is not found in the dictionary.

5.2.4 N-gram Handling

N-gram based matching is one of the approximate matching techniques. It used to handle words that are not found in dictionary (untranslatable words). In the technique, text strings are decomposed into n-grams, i.e., substrings of length n , which usually consist of the adjacent characters of the text strings. *Digrams* contain two and *trigrams*

three characters. The degree of similarity between the strings is computed on the basis of the number of similar n-grams and the total number of unique n-grams in the strings [25].

In this query translation system, digrams will be used for n-gram handler. Similarity values are computed using the following string similarity scheme:

$$\text{SIM}(N1, N2) = \frac{|N1 \cap N2|}{|N1 \cup N2|},$$

where N1 and N2 are digram sets of two words. $|N1 \cap N2|$ denotes the number of intersection (similar) digrams, and $|N1 \cup N2|$ the number of unique digrams in the union of N1 and N2 [25].

After similar words are found, dictionary translator will be used for translating them to English. Next, those translated words will be sent to English query constructor.

5.2.5 Target Language Query Structuring

After all translation has been done, translated or/and translatable words will go through this process. This process determines the query structure before it goes to the information retrieval part.

5.3 Data Design

As shown in Figure 5.3, the query translation system consists of two components which are stored in database. They are abbreviation list and dictionary. The reason of storing them in database is that it can reduce the response time. Dictionary look up using File-based dictionary will usually increase system response time because there are a lot of

entries inside the file and thus causing many loop for retrieve the entries. The system is designed as a web-based application, it will get worse if too many users access the file simultaneously and this kind of design will generate more clash.

Table 5.1 and table 5.2 define the data used for the stop word list and dictionary. The table for the bilingual dictionary is named *dictionary* while the table name for abbreviation list is *abbreviation*.

Table 5.1: Description of table *dictionary*

Field	Datatype	Description	
entryId	Int	ID of the dictionary entry (Primary Key)	Not Null
myWord	Varchar	Malay word entry	Null
enWord	Varchar	The meaning of the Malay word	Null
digram	Varchar	Digram set	Null

Table 5.2: Description of table *abbreviation*

Field	Datatype	Description	
abbId	Int	ID of the abbreviation (Primary Key)	Not Null
abbreviation	Varchar	abbreviation	Null
fullName	Varchar	Full name of its abbreviation	

5.4 User Interface Design

User interface design describes how the software communicates with the humans who use it. The web-based CLIR system user interface design is based on some of Human Computer Interface (HCI) design guidelines in general interaction, information display and data input. The design considerations focus on the effective general interaction between the user and the system, complete, unambiguous and easy understand into display as well as user-friendly data input medium.[24]

Consistency

The web-based CLIR system user interface design tasks considerations in the consistency of the interface for menu selection, command input, data display and etc. For example, a particular icon shape is used to represent a single meaning. The same icon shape that represents different actions or objects depending on the context will lead to confusion. The objects and operations provided are designed to form a minimum and consistent set so that the system is easy to learn and apply.

Minimizing Memorization

The operations in an interface are structured so that they are easy to understand and to remember. For example, using one key or button for all add operations is easier to remember than a number of different keys for different types of add operations.

Feedback

The system shows feedback to the user after a process is performed. Feedback is used to inform the user whether the process has been performed successfully.

Error Checking and Handling

The system presents itself from the errors that might cause it to fail. The system provides the error checking inputs and actions in most of the interactive applications.

Permit Reversal of Actions

The system web site allows the users to cancel or undo some actions in most of the interactive applications.

Error Messages

The design considerations also focus on providing the meaningful error message. The message should describe the problem that the user can understand. In addition, the message should provide constructive advice for recovering the error.

Minimize the Number of Input Actions Required of the User

The minimum amount of typing can be accomplished by using the mouse to select from the predefined sets of input. Besides, a "sliding scale" can be used to specify the input data across a range of values.

5.4.1 Web Page Design

Since the Malay-English CLIR system is a web-based system, the web page design considerations are taken into account. The web page design considerations are stated as the following: -

- Create a rich and rewarding experience for the users. Provide a reason for them to be glad to visit again the web site.
- Design an effective user interface to enable the users to be effective in accomplishing their tasks.
- Provide a common and consistent look and feel across the application. The web pages should reflect a consistent font, color, images, page background, and page layout.
- Give navigational cues to provide the proper guidance to the users in their journey. Make sure the users are informed where they are going during the navigation.
- Provide the user with a path at all the times. Do not create dead-end pages.

5.4.2 Screen Capture

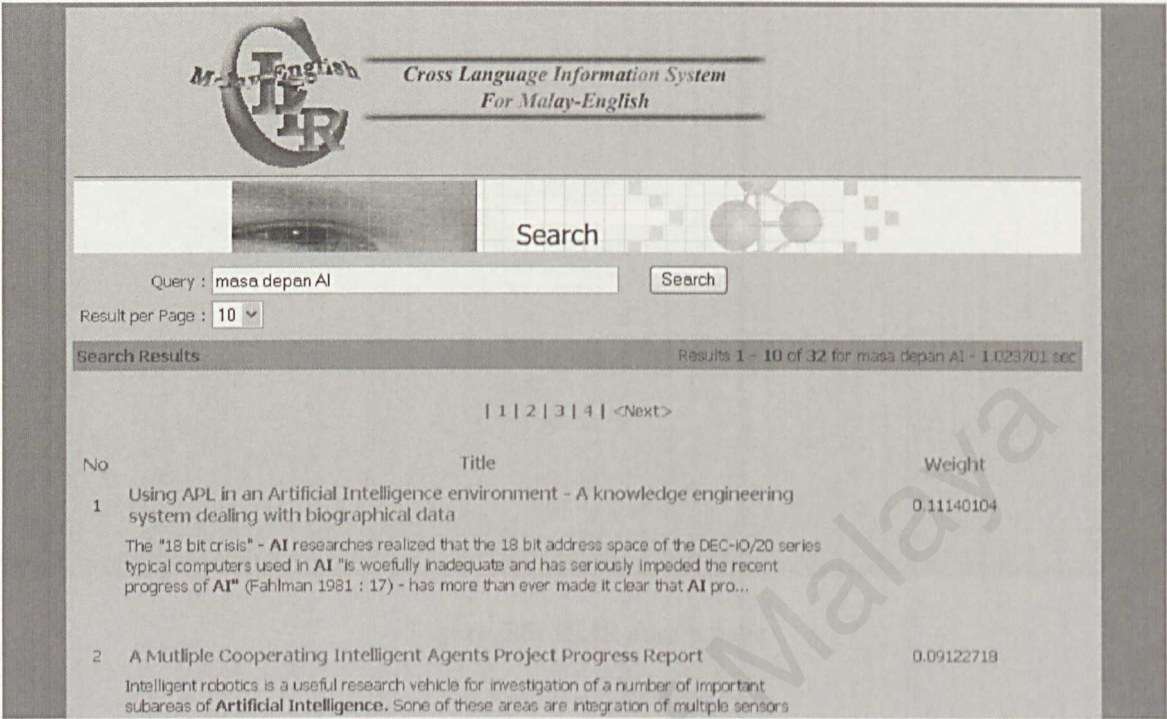


Figure 5.4: CLIR search page

Figure 5.4 shows the search page of the system. This page is the main part of the project where users can retrieve information through typing the Malay query in the text box.

Administrator is the person who maintains the system. Admin can add some other document by using the document indexing page (see Figure 5.5). Besides, Administrator can change threshold value in the same page.

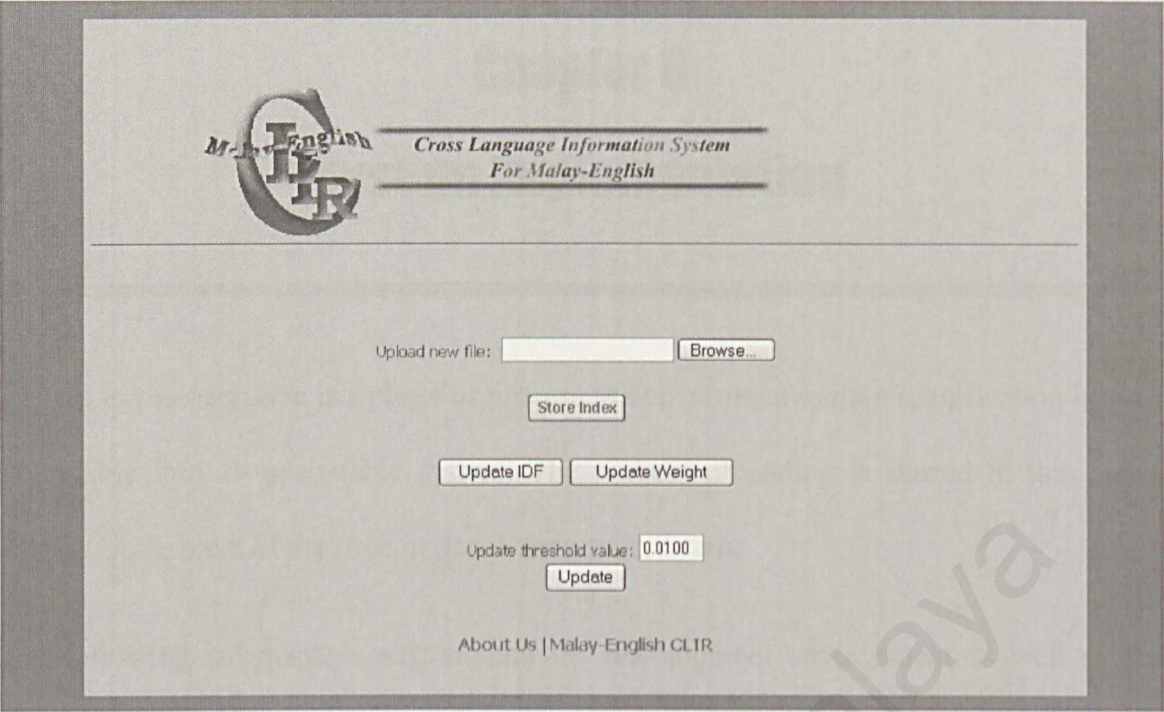


Figure 5.5: CLIR admin page

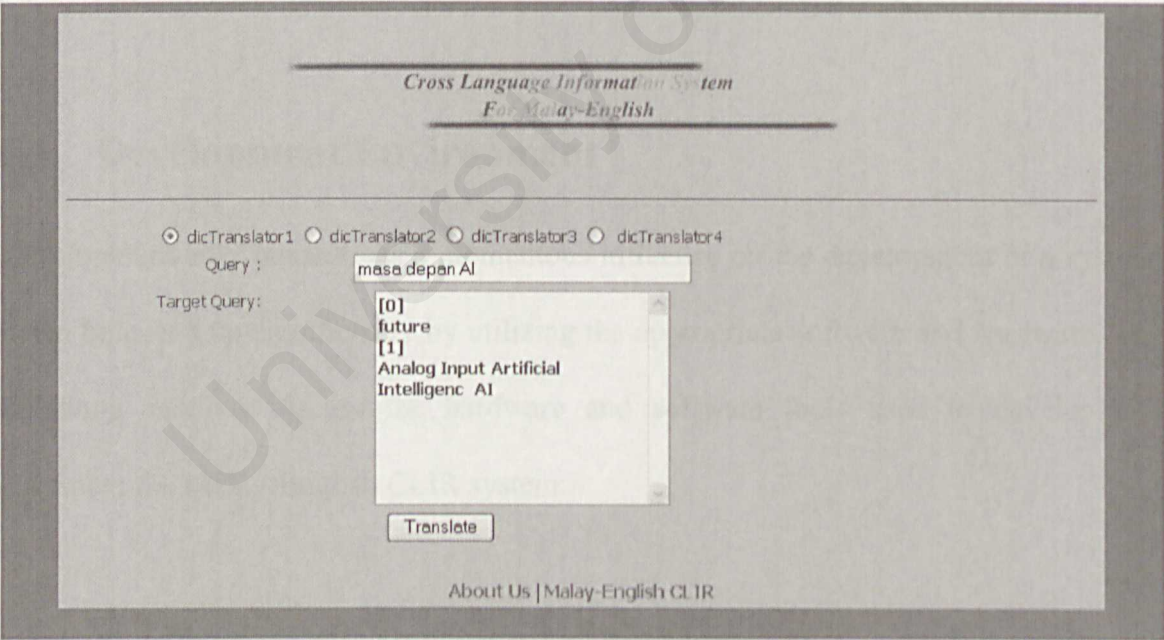


Figure 5.6: Dictionary translator page

Chapter 6

System Implementation

System implementation is a phase or process of converting a system specification into an executable into an executable system. The process of coding is started in this phase where it took most of the time in developing this system.

The following subchapters will explain the development environment as well as the development of the system itself, some system coding and the coding style and approaches that applied in the Query Translation System as well as the integrated Malay-English CLIR system.

6.1 Development Environment

Development environment has a momentous influence on the development of a system. It can be paced up significantly by utilizing the appropriate software and hardware. The following sections discuss the hardware and software tools used to develop and document the Malay-English CLIR system.

6.1.1 Hardware in the Development Environment

The hardware configured for the development environment is the underlying element of the whole system. The hardware used in the system implementation phase plays an important role in realizing the final system architecture.

The hardware configuration of the development environment is as below:

- Intel Pentium IV Processor 2.4GHz.
- Memory –512MB DDR SDRAM
- Storage – 3 GB of Hard disk space is reserved.
- Standard input and output devices
- Other standard desktop PC component.

6.1.2 Software Configuration

After hardware for the development environment is prepared, the first step is to install server and other related development tools. It is important to know the correct way to install an uncorrupted server so that the development can be done easier and without many obstacles. Below are the installation steps of the development server:

1. Install Microsoft Windows 2000 Professional
2. Install Apache HTTP Server (Appendix A)
3. Install PHP Engine (Appendix B)
4. Install MySQL Database Server (Appendix C)
5. Install MySQL Control Center (Appendix C)
6. Install Macromedia Dreamweaver MX
7. Install Zend Studio

6.2 Development of the System

6.2.1 System Development Process

Basically, the e-veterinary System is following a program development process that consists of 5 steps:

- i. Review the system documentation
- ii. Design the system
- iii. Code the system
- iv. Test the system
- v. Complete the system documentation

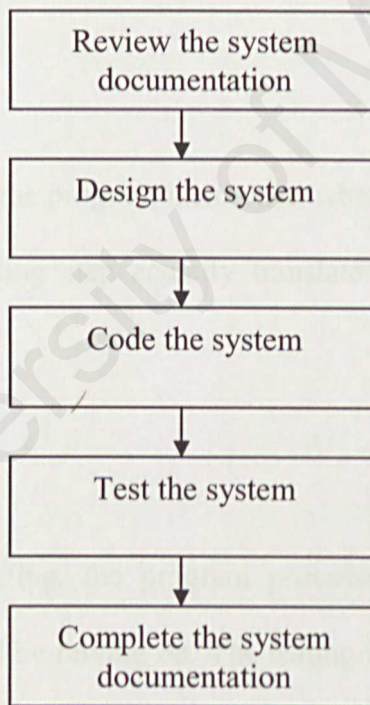


Figure 6.1: System development process

6.2.1.1 Review the system documentation

The program documentation that was prepared during the early phases needs to be reviewed. This documentation has the author to understand better of the work that need to be covered during the coding phase.

6.2.1.2 Design the system

After review the program documentation, the second level of program design needs to be completed during the system development where the author decides exactly what the program can accomplish. This is the process of what it must do by developing a logical solution to the programming problem.

6.2.1.3 Code the system

Coding is a process of writing the program instruction where this instruction implements the program design. The coding step actually translates the design specification to machine-readable format.

6.2.1.4 Test the system

During the level program testing, the program processes actual data and produces information on which user will be relying on. The testing involved most are unit testing and integration testing.

6.2.1.5 Complete the system documentation

Completing the program is essential for the successful operation and maintenance of the system. This documentation includes the system's user manual that may be needed by most of the customer as well as the system administrator.

6.2.2 Coding Principle Applied

Good programming skill helps in producing a good system and thus easier to maintain. There are a few principles need to apply when coding the program.

6.2.2.1 Readability

Readability is essential for future enhancement. Coding style and convention applied may strongly affect the readability. Codes need to be formatted to enhance understanding.

6.2.2.2 Reusability

Reusability is an important principle. It can be considered as a method for improving product quality throughout the system development process. In addition, it also reduces the coding time as well as the testing and documentation time.

6.2.2.3 Modularity

Software with effective modularity is easier to develop because function may be compartmentalized and interfaces are simplified. Independent modules are easier to

maintain because secondary effects caused by design or code modification are limited, error propagation is reduced, and reusable module are possible.

6.2.2.4 Good naming techniques

All name given to variables, controls, functions and modules should provide an easy identification to programmers. This naming process was done with the right code and standard.

6.2.2.5 Internal documentation

Internal documentation in programming code is important to help programmers to have more understanding with the program. Basically, it refers to internal comment, which is provided as a guideline to help programmers to understand especially in maintenance phase.

```
/**
 * Function Name : stem_list
 * Takes a list of words as parameter and returns them reduced to their stems.
 *
 * $words can be either a string or an array. If it is a string, it will
 * be split into separate words on whitespace, commas, or semicolons. If
 * an array, it assumes one word per element.
 *
 * return array List of word stems
 */

function stem_list( $words )
{
```

Figure 6.2: Comments in coding

6.2.3 Programming Method

Sub-systems in the system must be formed based on logical equitation, data requirements and sequences of function. Each sub-system must have at least one program. Coupling concept will produce modular programs, while binding concept will produce structured programs.

6.2.3.1 modular programming

Modular programming is a programming method that dividing a complex problem to a small parts to make easy to program. System is programmed using this method to avoid complexity problem and to make it easy to understand.

6.2.3.2 Structured programming

Structured programming is one way of regular and orderly programming. Coding steps that is used in this technique were as follow:

- Branching command without condition were abolished, or at least minimized its used, in each program modules.
- Commands in program routine must bas on logical sequence to ensure that there is only one source of input to routine and out of routine.
- Each routine must have complete codes with understandable comments.

6.3 Malay-English CLIR and Query Translation System

This thesis is more concern in the method for translating a Malay query to English Query and its system development. After the system has done, it will be integrated with Information Retrieval Module to success and complete the dictionary-based Malay-English CLIR system. This subchapter will discuss system directory structure, functions that applied in this system and some SQL statements that used for retrieving information from the MySQL database.

6.3.1 System directory structure

Before the development start, the site directory structure has been designed and organized for ease of maintenance. In general, all the files, which have the same purpose, will be put into the same directory. The images directory will store all the image files; these includes directory will store all include files; the db directory will store the files for database connection.



Figure 6.3: System directory structure

6.3.2 Functions used

Table 6.1: List of functions

Function Name	Description
dictranslator	<p>Parameter: user query (string)</p> <p>This function involves 5 processes:</p> <ol style="list-style-type: none"> query language identification dictionary lookup abbreviation checking ngram handling target language structuring <p>Return: structured English query (2 dimensional array)</p>
ngram	<p>Parameter: untranslatable word (string)</p> <p>The parameter is analyzed and similarities of the parameter with the entries in dictionary are counted and the entry(s) with highest similarity will be returned.</p> <p>Return: nearest words set (array)</p>
dictLookUp	<p>Parameter: untranslatable word (string)</p>

	<p>Translate a Malay word e.g. komputer, or compound word e.g. salahguna, or phrase e.g. masa depan, sains komputer.</p> <p>Return: translated words set (array)</p>
InsertDoc	<p>Parameter: document ID (int), title (string) and full text (string)</p> <p>Insert all three parameter into table <i>documents</i>.</p> <p>Return: result of the SQL execution (boolean)</p>
InsertData	<p>Parameter: Index term (string), frequency (double), IDF (double), weight (double), file ID (int)</p> <p>Insert all three parameter into table <i>indexes</i>.</p> <p>Return: result of the SQL execution (boolean)</p>
countInvert	<p>Parameter: None</p> <p>Update the inverted document frequency of all index terms in table <i>indexes</i>.</p> <p>Return: result of the SQL execution (boolean)</p>
CountWeight	<p>Parameter: None</p> <p>Update weight of all index terms in table <i>indexes</i>.</p> <p>Return: result of the SQL execution (boolean)</p>
retrieval	<p>Parameter: translated query (string)</p> <p>Retrieve or search the relevant documents by using the translated query.</p> <p>Return: retrieved documents (array).</p>
getThreshold	<p>Parameter: None</p> <p>Send SQL statement to retrieve threshold value.</p> <p>Return: threshold value (double)</p>
stemming	<p>Parameter: a set of words (string)</p> <p>Perform stemming process for all words in the word set.</p> <p>Return: stemmed word set (array)</p>
boldKey	<p>Parameter: full text of retrieved document (string), translated English query (string)</p>

	<p>High light all words in full text of retrieved document which appear in translated English query.</p> <p>Return: full text with bold words (string)</p>
array2str	<p>Parameter: array and total left most pop, n</p> <p>Combine all elements in array to become a string where the string is without the n left most words.</p> <p>Return: string without the n left most words (string)</p>
randValue	<p>Parameter: None</p> <p>Generate a random value</p> <p>Return: random value</p>

6.3.3 SQL statements

There are several common SQL statements that used in the Query Translation System:

For query language identification,

```
SELECT enWord, myWord FROM dictionary
```

For dictionary lookup,

```
SELECT * FROM dictionary
```

For abbreviation checking,

```
SELECT * FROM abbreviation
```

For Ngram handling,

```
SELECT * FROM dictionary WHERE ngram != NULL
```

Chapter 7

System Testing

Testing is critical in uncovering logical error and to test the system reliability. The main objective of testing is to uncover different types of errors that exist while executing the system. System testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. However, testing can only show that software defects are present.

7.1 Type of Faults

Types of faults that can be found in the system are algorithmic faults, throughput faults and computation faults:

7.1.1 Algorithmic Faults

Algorithmic faults use to happen when a component's algorithm or logic fails to produce the expected output for a given input. This kind of faults is always due to wrong proceeding steps.

7.1.2 Throughput/Performance Faults

Throughput faults or performance faults use to happen when the component does not produce an expected speed. Some of these faults are caused by inefficient of coding or

poor manipulation of SQL statements. When discover this fault in the system, the fault is being carefully observed and monitor continuously to ensure performance meet requirement. In most cases, the fault occurs during massive usage of database connection.

7.1.3 Computational/Precision Faults

Computation or precision faults are occurred when a particular formula is being implemented wrongly or the component does not complete a result to a required accuracy. These faults are being tested in component such as N-gram handler. In this system, all components are being checked and tested with dummy data to ensure the computation works accordingly with its situation.

7.2 Testing Process

In developing a system, testing usually involves several stages. An example of testing process is shown as below:

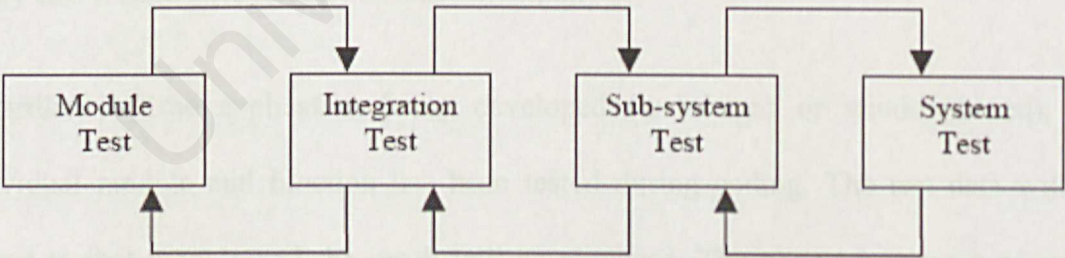


Figure7.1: Testing process

There are several sub-modules need to be developed in other to complete the module of query translation system. Since there is more than one subsystem need to be tested, a well designed flow of testing is needed to ensure the integrity of the entire Malay-English CLIR system. In this thesis, a hybrid of bottom-up and top-down approaches has been used to make the testing for the entire system more effective and efficient.

The bottom-up approach has been used from the beginning till the end of development. Whereas the top-down approach has been used after then entire system has been completely developed. The testing will then be carried out from top-level and narrow down to the lowest level.

7.2.1 Module Testing

Query translation module consists of several sub-modules. During the development, these small modules are tested to ensure that they function correctly. The module testing is always done in a controlled environment. So that, the predetermined flow of actions can be taken on the module. The result produced by the module will be observed to verify that it function correctly in most situation.

Regardless of the application being developed (web-based or window-based), the individual module and function has been tested during coding. The test data will be passed to that module and the result will be observed. The common sources of errors found are: -

- i. Improper or inconsistent typing
- ii. Erroneous initialization or default values

- iii. Incorrect variable names
- iv. Inconsistent data types
- v. Overflow, underflow, address exceptions.

7.2.1.1 PHP script debugging

The system is designed as a web application, thus testing is always a complicated and time-consuming task due to the lack of clear and specific error message. Therefore, there are several ways to simplify the debugging method:

- i. First, backup a copy of testing file. This is because, when testing take place, it is too easy to overwrite or delete necessary piece of code accidentally.
- ii. Attempt to figure out which lines causing the error. Isolate the code by comment out those lines of code. Then, refresh the browser and see whether the error still exists.
- iii. Print out the suspected variable and check whether the value is correct.

7.2.2 Integrated Testing

When one of the module has been fully tested, it will then be tested along with other modules, namely integration testing. This is to ensure that the interfaces among the modules are defined and handled properly.

The incremental integration has been used for testing. It means that there will be more modules integrated together for testing from time to time as most of them are developed

towards the end of development. However, the critical modules have been given priority to be tested and integrated early. The reusable or shared components are considered as critical modules. Basically, the integration test will be focused on these issues: -

- i. Interface integrity
- ii. Functional validity
- iii. Information content
- iv. Performance

7.2.3 System Testing

After query translation system and information retrieval system are integrated into one system, the system testing is taken place. A system testing is a series of different test designed to fully exercise the system to uncover its limitation and to measure its capabilities. The objective is to test an integrated system and verify that it meets the specified requirements.

7.2.4 Testing Check List

Below is the testing check listing that done during the testing process:

Table 7.1: Testing check list

<u>Testing</u>	<u>Observation</u>	<u>Error</u>	<u>Result</u>
1. Language Identifier			
Insert a query where Malay words more than English words	Return Malay query	No	Passed
Insert a query where English words more than Malay words	Return English query	No	Passed
Insert a query where English words equal to Malay words	Return English query	No	Passed
2. Dictionary Lookup			
Put no input and type submit button	All entries in dictionary are retrieved. Solved by using JavaScript to control the input.	Yes	Fixed
Insert a misspelled word	Result not found	No	Passed
Insert <i>pencetak</i>	Return "printer"	No	Passed
3. Ngram Handler			
Put no input and type submit button	Message is prompted to ask user to enter an input.	No	Passed
Insert a misspelled word – <i>peencetak</i>	Return promising result – <i>pencetak</i> (similarity = 0.875) Similarity = $7/8 = 0.875$	No	Passed
Insert " <i>pencetak</i> "	Return " <i>pencetak</i> "	No	Passed
4. Abbreviation Checker			
Put no input and type submit button	Message is prompted to ask user to enter an input.	No	Passed
Insert a misspelled word	Result not found.	No	Passed
Insert " <i>pencetak</i> "	Return "printer".	No	Passed

6. Query Translator System			
Put no input and type submit button	Message is prompted to ask user to enter an input.	No	Passed
Insert a sentence with many white space between each word	Return no result. Solved by removing extra white spaces.	Yes	Fixed
Insert a sentence with misspelled word in it	Return promising result (all words including misspelled word are translated)	No	Passed
Insert abbreviation	Return full name of the abbreviation.	No	Passed
Insert English query	Return the original English query.	No	Passed
Record respond time	Less than one second.	-	-

Chapter 8

System Evaluation

Evaluation is the ultimate phase of developing a system and an important phase before system is delivered. System evaluation is implemented by more than simply comparing the information obtained with the information which is expected. It was related to user environment, attitudes, information priorities and several other concerns that are to be considered carefully before effectiveness can be concluded. At all phases of the system approaches, evaluation is a process that occurs continuously, drawing on a variety of sources and information.

8.1 Evaluation on Query Translation System

8.1.1 Evaluation on Applied N-gram Handler

As discussed in chapter 2, common Cross-Language Information Retrieval system cannot solve the problem of untranslatable words. One of the solutions is to return the untranslatable word set to its query construction process. However, there is another more efficient and reliable way to solve this problem – N-gram method. In this thesis, di-gram method is implemented and it gives a good result.

Two di-gram handlers have been developed and the only difference is the number result returned – one returns the nearest word and the other one returns 5 nearest words.

Table 8.1 shows the result return by the e two N-gram handlers with a set of misspelled Malay words.

Table 8.1: Evaluation of 2 N-gram handlers

Misspelled Malay word set	Result of first N-gram handler	Result of second N-gram handler
computer	komputer	komputer minikomputer makrokomputer supercomputer putera
peencetak	pencetak	pencetak mencetak telepencetak cetak petak
titikus	tikus	Tikus itik tetikus titik titi
moniter	monitor	Monitor montok ton mnemonik tong

From the result above, the second N-gram handler includes all expected result. However, it holds too many irrelevant result and this will eventually resulting the IR system to retrieve far more many irrelevant documents. Therefore, the first N-gram is chosen for the Malay-English CLIR system.

8.1.2 Evaluation of Query Translation System

The quality of a Query Translation system determines the quality of the whole CLIR system. In this section, the performance of the developed Query Translation system will be discussed. Below are 5 query set that used to test its performance:

- i. *Masa depan AI* (the future of AI)
- ii. *Teknologi kembara terkini* (the latest mobile technology)
- iii. *Isu keselamatan dalam perdagangan elektronik* (security issue in E-commerce)
- iv. *Kebaikan dan keburukan Internet* (advantages and disadvantages of the Internet)
- v. *Robot pintar dan penglihatan* (intelligent robot and vision)

Table 8.2: Evaluation of implemented Query Translation System

Query	Result
<i>Masa depan AI</i>	[0] future [1] Analog Input Artificial Intelligence AI
<i>Teknologi kembara terkini</i>	[0] technology [1] explore wander [2] now terkini
<i>Isu keselamatan dalam perdagangan</i>	[0] to meet to encounter to join Isu [1]

8.1.3 Recall and Precision	safe south keselamatan [2] inside in deep [3] trade commerce
<i>Kebaikan dan keburukan Internet</i>	[0] goodness benefit [1] and [2] ugliness badness wickedness [3] to nag Internet
<i>Robot pintar dan penglihatan</i>	[0] robot [1] cunning sharp-witted smart clever [2] and [3] displacement penglihatan

Table 8.2 shows that some of the queries are translated very well. However, there are still consisting untranslatable words even though all the words in user queries do not spelled mistakenly. This is due to the limited entries in the used Malay-English dictionary. These results will eventually affect and reduces the recall and precision of the whole Malay-English CLIR system.

8.1.3 Recall and Precision

For this section, query set in section 8.1.1.3 will be reused. The query set will be tested on the Malay-English CLIR system. Firstly, the Malay queries of the query set are being used to retrieve documents from a document collection. The recall and precision for each retrieval process is recorded and the recall-precision curve is shown in figure 8.1.

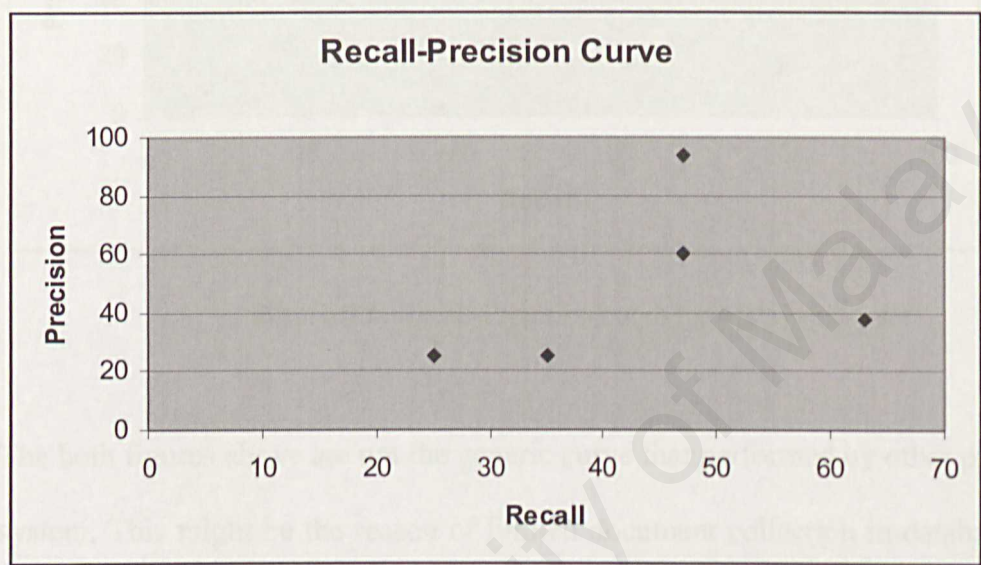


Figure 8.1: Recall-Precision curve for Malay query set

After that, the English query set is also used to plot the recall-precision curve below.

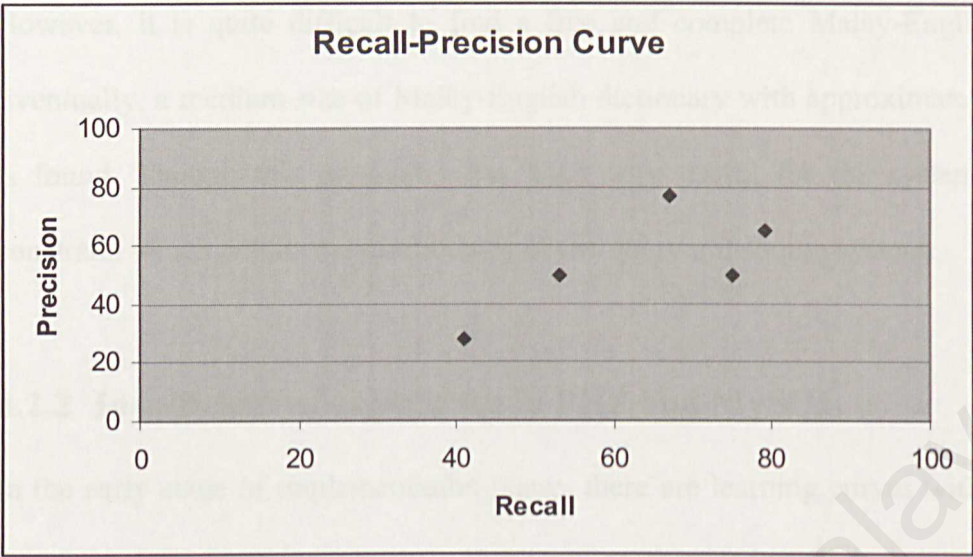


Figure 8.2: Recall-Precision curve for Malay query set

The both figures above are not the generic curve that performed by other professional IR system. This might be the reason of limited document collection in database. However, the system provides a promising result and it should be able to get a better result if the enhancements suggested in latter section are implemented.

8.2 Difficulty and Suggested Solution

The success of the Query Translation system does not come without problems and constraints. Much effort had been placed into understanding, eradicating, and solving problems encountered through the whole course of the system development.

8.2.1 Problem in finding a better Malay-English dictionary

In order to complete the Query Translation system, a Malay-English dictionary is needed. However, it is quite difficult to find a free and complete Malay-English dictionary. Eventually, a medium size of Malay-English dictionary with approximate 14000 entries is found. Though this dictionary has been very useful for the system, it is still a constraint which results the inefficiency of the query translation system.

8.2.2 Insufficient of experience in PHP and MySQL

In the early stage of implementation phase, there are learning curves with mastering to the chosen programming language. Furthermore, there is insufficient time to pick up the language. This result the learning and developing process was done in parallel. Although having difficulty in the early stages, however choosing to program in PHP proves to be a wise move as it is a very powerful technology to build a web based application. Problems were solved through research on related material online and referring to some reference books.

8.2.3 Time Constraint

The project time frame is very short and tight, as all the development and documentation need to be done in less than 4 months. Since there are assignments and lectures, the time allocated for the development process is very short. Furthermore, the allocated time is fragmented, as necessarily for attending lectures in the middle.

8.3 System Strength

8.3.1 Acceptable for Both English and Malay query

The system consists of a query language identifier which able to differentiate the query language. In other words, users are not restricted to use only Malay language.

8.3.2 Manageable Threshold Value

The system does not retrieve documents which have a very low weight. The filtering is done by giving itself a threshold value. This means index term with lower weight than the threshold value will not be retrieved.

This threshold value is changeable. Admin of the system is the person who can modify the threshold value. This feature makes the system more flexible and useable.

8.3.3 Systematic Error Handling

User will get an error message if he or she submit a bad input or the system itself encounter technical problem. This let user more understand how the system works and the user will prevent themselves to repeat the action.

8.3.4 Easy to Use

It is relatively easy for the user to use this system as no much complex linkages. This lets users know easily how the system functions.

8.3.5 Nice interface design

The system integrated with a good and aesthetical interface template. With the CSS design and some graphics, users will feel more comfortable to stay around the page.

8.4 System Limitation

8.4.1 Limited document collection

Due to the time constraint, system does not have enough documents in its database. Furthermore, the scope of the project is restricted in the field of computer science. Thus, the system is not yet ready for going “alive”.

8.4.2 Limited entries of dictionary

As discussed in section 8.1.2.1, we need a more complete dictionary to get a more efficient and reliable Query Translation system. Due to its limited entries, target language query are not 100% usable for retrieving information in the database.

8.4.3 Slow indexing process

The respond time of the indexing process is slow. This is due to insufficient memory and CPU speed. In order to address this problem, a server with better hardware specification are required for faster processing speed.

8.4.4 One Direction Method

System allows users to use both English and Malay language as input query yet the retrieved documents are all in English. In other words, the system is Malay to English CLIR system but not English to Malay CLIR system.

8.5 Future Enhancement

Due to the limitation of this system, there are a few suggestions that may be useful to future enhancement of the Malay-English CLIR system. The suggestions are as below:

8.5.1 More Documents Added

It is very important to get as much documents as possible for the document collection. It does not make sense if the search engine searches documents in a database which has only few documents. Therefore, the database should store as much documents as possible such that the CLIR system can search more relevant documents.

8.5.2 Use A Good Dictionary

In order to make the system more usable, the query translation system needs a complete Malay-English dictionary. This dictionary should be reliable and consistent in the sense of its correctness of translation e.g. Dictionary from *Dewan Bahasa dan Pustaka*.

8.5.3 Better Method of Indexing Process

The current used method of indexing process is quite inefficient. The process is slow as it uses plenty of processes during the database insertion. Some other methods need to be studied for the modification (e.g. compression or other methodology).

8.6 Knowledge and Experience Gained

Besides knowledge on technical aspects such as Windows 2000 Server, PHP, web technologies and MySQL Server, there are also other valuable experiences gained from working on this project such as:

- i. Being exposed to the real system development environment especially dealing with users
- ii. Learn how to manage a project as in time and resource
- iii. Concept on how to integrate and fully utilize various technologies into developing system
- iv. Experience on how to set up and configure various technologies to be able to serve as a live system.
- v. Learn to work independently
- vi. Cultivated skills in writing documentations and reports
- vii. Boost self-confidence, self-esteem and good communication skill

8.7 Review on Goals

There should be certain expectation and objective achieved at the final stage of the project.

8.7.1 Expectation Achieved

The system had fulfilled the expectation stated at the early stage of the project. All the basic foundation of the system was being designed and implemented. Moreover, the end product met the criteria such as user friendliness, reliability, manageability, expandability and so on.

8.7.2 Objective Achieved

The system created had fulfilled all the requirements stated in the early chapter, therefore, the objectives to establish the application had been achieved.

References

- [1] Hayashi, Y., Kikui, G. and Susaki, S. (1997) Titan: A cross-linguistic search engine for the www. In *AAAI Symposium on Cross-Language Text and Speech Retrieval*. American Association for Artificial Intelligence.
- [2] *Cross Language Information Retrieval: A Research Roadmap*, a workshop at SIGIR-2002: 22nd International Conference On Research And Development in Information Retrieval, Aug 2002, Tampere Finland.
- [3] Carol Peters and Eugenio Picchi. *Across languages, across cultures: Issues in multilinguality and digital libraries*. D-lib magazine, The Magazine of Digital Library Research, May 1997, Available: <http://www.dlib.org/dlib/may97/peters/05peters.html>
- [4] Demetriou G., Keskustalo H., Sepponen B., Herring P., Franzén K., Karlgren J., Olsson F., Gaizauskas G. and Sanderson M. (2003). "A Web Services Architecture for Distributed Cross-Language Information Retrieval".
- [5] GBdirect, "Active Web Sites and Comparison of Scripting Languages", [Online], available:http://training.gbdirect.co.uk/courses/php/comparison_php_versus_perl_vs_asp_jsp_vs_vbscript_web_scripting.html [10 Aug 2004]
- [6] Edelstein, Herb. "Unraveling Client/Server Architecture." DBMS 7, 5 (May 1994): 34(7).
- [7] Schach, S.R. Object-Oriented and Classical Software Engineering, 6th edition, USA: McGraw-Hill.

- [8] Fujii A., Ishikawa, T. (2000) *Applying Machine Translation to Two-Stage Cross-Language Information Retrieval*. University of Library and Information Science, Japan. Available: http://arxiv.org/PS_cache/cs/pdf/0011/0011003.pdf
- [9] Peters C., Sheridan P. (2002) *Cross-Language Information Retrieval: Recent Trends*. Conference Proceedings, 2002.
- [10] <http://www.hsl.creighton.edu/hsl/Searching/Recall-Precision.html>
- [11] Sundheim B., *Task-Oriented Text Analysis Evaluation*, USA, Available: <http://cslu.cse.ogi.edu/HLTsurvey/ch13node4.html>
- [12] Hedlund T. (2003) *Dictionary-based Cross-Language Information Retrieval: Principles, System Design and Evaluation*. University of Tampere.
- [13] Pirkola A., Hedlund T., Keskustalo H., Jarvelin K. *Dictionary-Based Cross-Language Information Retrieval: Problems, Methods, and Research Findings*. Information Retrieval 4(3/4), 209-230, University of Tampere.
- [14] *A comparison of Operating system*. (2002) [online], Available: <http://www.dreamwater.org/prip/comparison.html>
- [15] Phil Hughes, *Operating Systems Comparison*, [online], Available: <http://www.netshooter.com/linux/oscomp.html>
- [16] David, Pitts., *Red Hat Linux 2nd Edition, 2nd ed.*, Sams and Macmillan, 1998, United State of America.
- [17] *Database: an evaluation & comparison*, Bloor Research, 2001.
- [18] (<http://www.microsoft.com/>), 17/8/2004.


- [19] (<http://www.linux.org/info/index.html>), 17/8/2004.
- [20] (<http://www.redhat.com>), 17/8/2004.
- [21] (<http://www.apache.org>), 17/8/2004.
- [22] (<http://www.mysql.com>), 18/8/2004.
- [23] (<http://www.oracle.com>), 18/8/2004.
- [24] Hearn, D. and Bakar, M.P. , Computer Graphics : C Version, 2nd edition, New Jersey : Prentice-Hall International, Inc. , 1997, pp.271-276.
- [25] Pirkola, A, Keskustalo, Heikki, Leppänen, Erkka, Käsälä, Antti-Pekka and Järvelin, Kalervo (2002) "Targeted s-gram matching: a novel n-gram matching technique for cross- and monolingual word form variants." Information Research, 7(2), [Online], Available: <http://InformationR.net/ir/7-2/paper126.html>

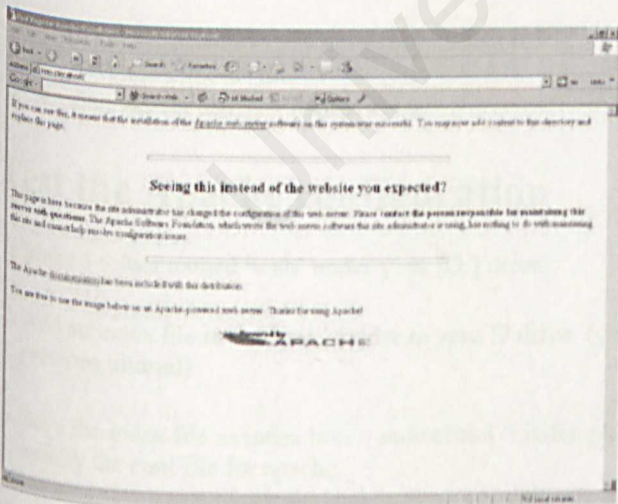
Appendix A: Install Apache

1. Double click "apache_2.0.52-win32-x86-no_ssl-installer" in the Installation folder which you have just downloaded and unzipped.

- Welcome to installation wizard for Apache HTTP Server 2.0.52
- Next >
- Check "I Accept" – Next >
- Next >
- Server Information :
 - Network Domain: Domain Name, Workgroup, otherwise use 'User'
 - Server Name: localhost
 - Administrators Email: your email address – Next >
 - For All Users on Port 80 as a Service – Recommended
- Setup Type: Typical :: Next > :: Next > :: Install :: Finish

Configure Apache

1. Remember when Apache was installed, on 'Server Information' we specified localhost as 'Server Name'. Apache was configured just to run on default.
2. Apache starts automatically after install. You will see this icon in the bottom right hand side of the screen.
3. Now that Apache has been configured, let's try it out. Open your web browser and type <http://localhost>



4. You should see this after you have typed the address:

'If you can see the above screen, it means that the installation of the Apache web server software on this system was successful.'

5. Well, let's make a little index file for apache to use every time you are in `http://localhost`

- First, open the configuration file(`httpd`) - located in the '`conf`' folder of Apache
 - Start > All Programs > Apache HTTP Server 2.0.52 > Configure Apache Server > Edit ...
- Search for **DirectoryIndex** then we will add little extra characters to that line.
- Follow the image below. just add `index.php` and `index.htm` after `index.html.var`

```
# is requested.
```

```
# The index.html.var file (a type-map) is used to deliver con  
# negotiated documents. The MultiViews option can be used fo  
# same purpose, but it is much slower.
```

```
DirectoryIndex index.html index.html.var index.php index.htm
```

- Search for **DocumentRoot** and then modify

```
# DocumentRoot: The directory out of which you will serve your  
# documents. By default, all requests are taken from this directory, but  
# symbolic links and aliases may be used to point to other locations.
```

```
DocumentRoot "C:/Program Files/Apache Group/Apache2/htdocs"
```

← Remove this line and add the following

```
DocumentRoot "D:/web"
```

- Search for **Directory** and then modify

```
# This should be changed to whatever you set DocumentRoot to.
```

```
<Directory "C:/Program Files/Apache Group/Apache2/htdocs">  
<Directory "C:/web">
```

← Remove this line and add the following

- Save and Close the file.
- Stop Apache and Start Apache again.

Test the Apache configuration

1. Make a folder named 'web' under your [D:] drive.
2. Add an index file in the 'web' folder in your D drive. (you can use any html file you created in the previous tutorial)
3. Save the index file as `index.htm` :: `index.html` :: `index.php` or whatever extension you want as long as you modify the conf file for apache.
4. Type `http://localhost/` and you will see the index loaded in your browser

Appendix B: Install PHP

1. Unzip the folder "php-5.0.3-Win32", rename it to "PHP"
2. Copy and paste into C:\

Configuring PHP

1. Go into folder 'C:\PHP\ find file 'php.ini-recommended'
2. Rename this file to 'php.ini' open this file, look for "extension=php_mysql.dll", remove the ';' in front of this statement.
3. Copy 'php_mysql.dll' from folder 'ext' to C:\PHP\

Configure PHP for Apache

1. Open the 'httpd.conf' file that we opened earlier configuring Apache.
"Start->Programs->Apache HTTP Server 2.0.47->Configure Apache Server-> Edit the Apache httpd.conf Configuration File"
2. Go to the line or search for 'AddType application/x-tar .tgz'. Below that line add the following:
-You can copy and paste them.

```
ScriptAlias /php/ "c:/php/"  
AddType application/x-httpd-php .php  
AddType application/x-httpd-php .php3  
AddType application/x-httpd-php .htm  
AddType application/x-httpd-php .html  
Action application/x-httpd-php "/php/php-cgi.exe"
```

3. Save and Close the file.

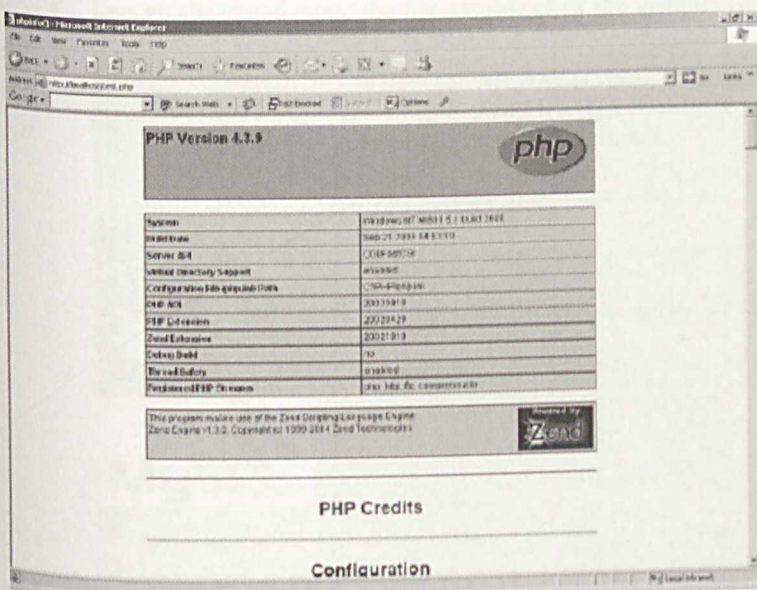
Testing PHP

1. If your apache was still running while you configured php then shut it down and restart it.
2. Lets go ahead and test php with a simple line.
3. Open Notepad by clicking 'Start' > Programs > Accessories > Notepad
4. Type in (don't forget to include <? and ?>)

```
<? echo phpinfo() ?>
```

5. Save the file under 'c:\web\test.php'
6. Open your browser and type this at the location box 'http://localhost/test.php'

You will see the following screen:



Appendix C: Install and configure MySQL server, Admin, Control Center

1. Double click "mysql-4.1.7-essential-win" in the Installation folder which you have just downloaded and unzipped.

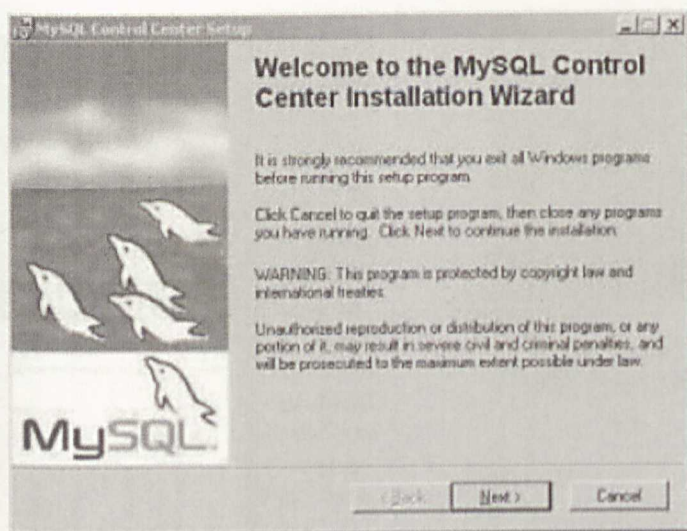
- Welcome to Setup Wizard for MySQL Server 4.1...
- Next > Click "Complete" > Next > Install
- Sign in for a MySQL.com account if you do not have one.
- Type in required information and click next.
- Tick the MySQL Configuration now > Finish
- Welcome to the MySQL Server Instance Configuration Wizard 1.01
- Next > Click "Standard Configuration"
- Next > Install As Windows Service > Uncheck "Launch the MySQL Server automatically"
- Next > Type in your root password > Check "Root may only connect from the localhost"
- Next > Execute > Finish

2. Double click "mysql-administrator-1.0.14-win" in the Installation folder which you have just downloaded and unzipped.

- Welcome to Setup Wizard for MySQL Administrator 1.0...
- Next > Click "I Accept the" > Next > Next
- Click "Complete" > Next > Install > Finish

3. Unzip the folder "mysqlcc-0.9.4-win32"

- Lets go ahead and unzip the file, double click the installer [mysqlcc or Setup].

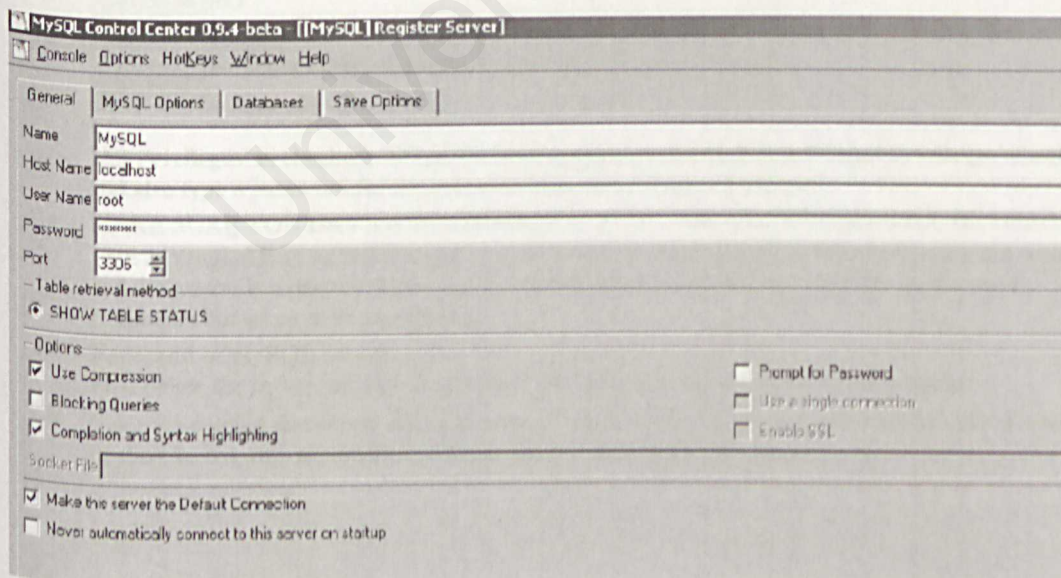


Next -> "I Accept:" Next -> :: Next -> :: Next -> :: Next -> :: Next -> :: Finish

Open MySQL

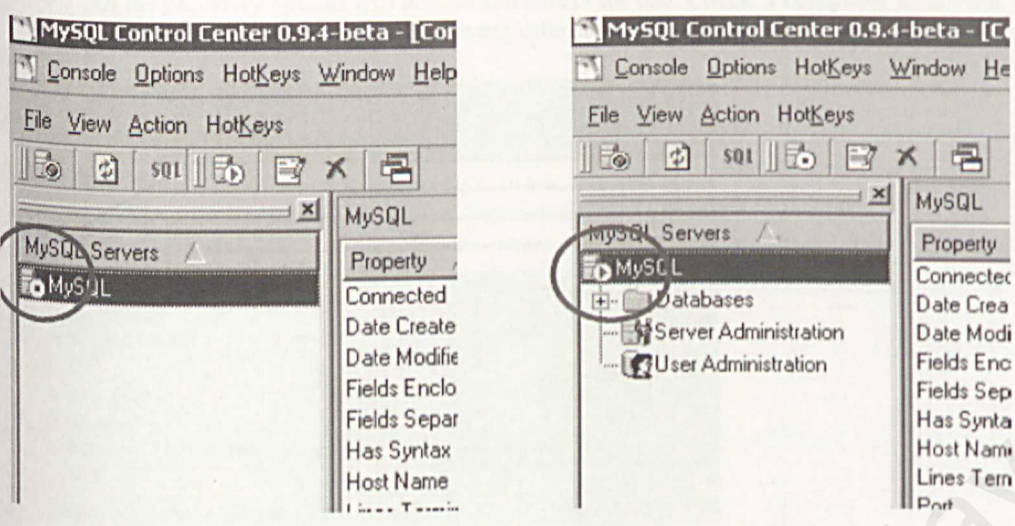
1. Start MySQL Control Center

- Start -> All Programs -> to start MySQL Control Center
- At first you are prompted with a dialog to register an existing server.
- Fill the dialog box fields with the following information provided by the image then click 'Add'



- Connection added successfully
- Connect to the server added

2. Double click on 'MySQL' to Query the database. It turns immediately to a green triangle where there is a red pixel.



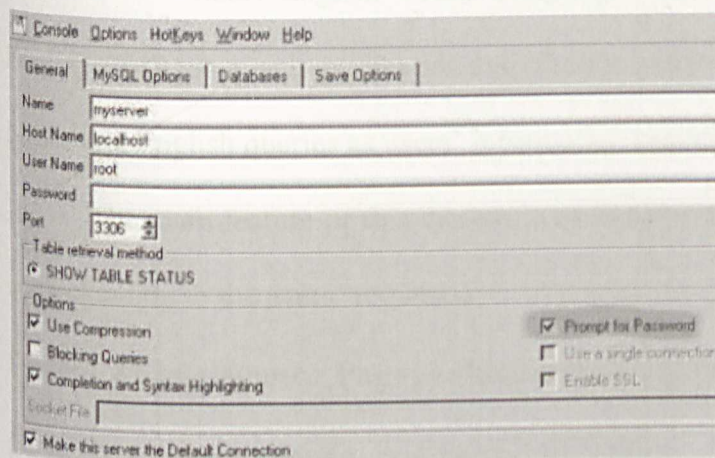
3. Configuring users and permissions

- Double click on 'User Administration' to see all the users that can access the server.
- So what you have at first is what the image below shows. We are going to have only one user and only one. Its going to be 'root' with a password which will be used in the *test script*.
- Right click on 'User Administration' and select 'New User' and fill using this data.
Make sure you have a password!

Console Options HotKeys Window Help	
Username	root
Host	localhost
Password	

- 'All Privileges' is checked, 'With GRANT option' is checked, and 'Global Privileges' checked and also type a password for the account! then click 'Add' then Close
- **MAKE SURE YOU HAVE A PASSWORD!!!! AND THAT PASSWORD WILL BE USED FOR THE MYSQL TEST SCRIPT.** I get a lot of people getting errors that their scripts are not working and the reason is a password that is not added in the test script. It says clearly on the script comments but some users just miss it.
- **Redo add of mySQL server**
First delete the server 'mysql'. Right click 'mysql' then delete and close the program
- Since we deleted that server only the account 'root' with your 'password' will be able to work or in short be the only account to give you access to mySQL there after.

- Now, let's redo the addition of MySQL server again. Open mysqlcc. Immediately you are prompted with the 'Register Server' dialog inside of mysqlcc.
- All the necessary options will be checked except for one. Check 'Prompt for password' and fill out the rest of the box with the following information then click 'Add'

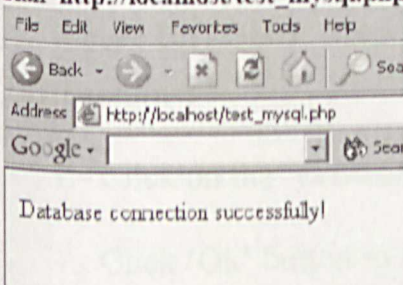


- Double click on 'myserver' you will be prompted for a password to connect to the database. The password is the one you set up earlier... just type it in and you are immediately connected.
- Drop to 'User Administration', delete all the users except for 'root@localhost'. Close the program and open it again. You are prompted with the password... type it in and you are connected to MySQL and you have only one user available to access MySQL and that user is 'root' alone.
- Cut and Paste the following code into 'notepad' and save it as 'c:\web\test_mysql.php'

```
<?php
//connect to the database server
$db = mysql_connect("localhost", "root", "password");

//report the connection failure or success
if (!$db) {
    echo "There was a problem connecting to the database.";
    exit;
}
if ($db) {
    echo "Database connection successfully!";
    exit;
}
?>
```

- Run 'http://localhost/test_mysql.php'



APPENDIX D: USER MANUAL

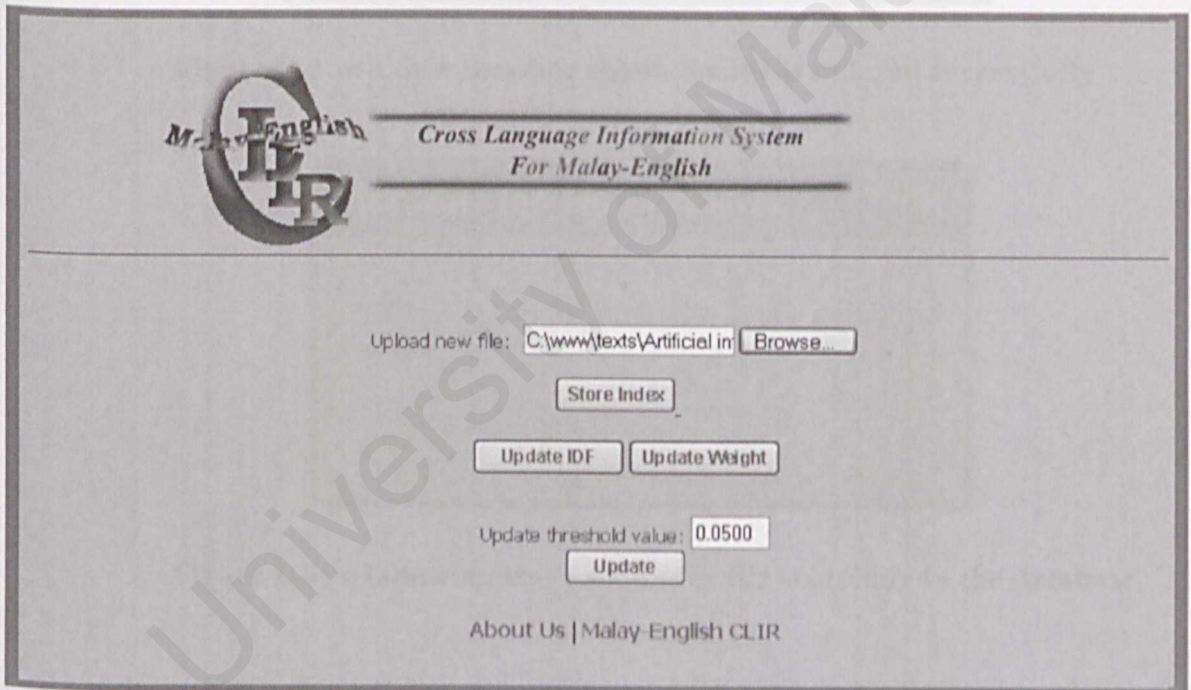
Introduction

Malay-English Cross Language Information Retrieval System (Malay-English CLIR) is a web application information retrieval system that able to accept both Malay and English queries as users' information request.

The main feature of this system is to retrieve the most relevant documents that is corresponding to the users' requests.

System Administrator Page: - Document Preprocessing and Indexing

(i) Document preprocessing and indexing



The screenshot shows a web interface for the Malay-English Cross Language Information System. At the top left is a logo with the letters 'M', 'E', and 'CLIR' in a circular arrangement. To the right of the logo, the text reads 'Cross Language Information System For Malay-English'. Below this, there is a section for file upload and indexing. It includes a text input field for 'Upload new file:' with the path 'C:\www\texts\Artificial in' and a 'Browse...' button. Below the input field is a 'Store Index' button. Further down are two buttons: 'Update IDF' and 'Update Weight'. Below these is a text input field for 'Update threshold value:' with the value '0.0500' and an 'Update' button. At the bottom of the interface is a link that says 'About Us | Malay-English CLIR'.

Figure D.1: Quick Document Preprocessing and Indexing Page

Instructions:

1. Click on the 'Browse' button and choose the file you wish to process.
Click 'Ok' button to choose the file. Make sure the chosen file is in the format of text file.

2. Then, click on the 'Store Index' button. When the indexing process is done, the system will alert you with a window message as shown in Figure D.2. If the selected file is processed, the system will alert you with another window message as shown in Figure D.3

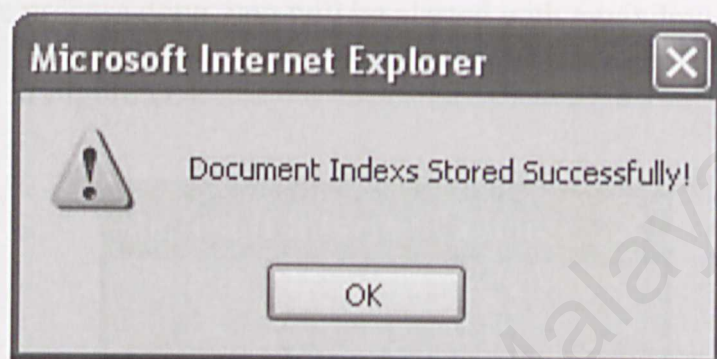


Figure D.2: window message shows the file is indexed successfully



Figure D.3: window message shows the file is already in the database

3. Browse another file you wish to process and repeat Step 2.
4. Before leaving the page, you must click on 'Update IDF'. 'Update Weight' button. If not, the inverted document frequency and weights are not updated. As so, you may cause the indexing process not completely done.

5. Firstly, click on 'Update IDF' button and wait for it to process completely. Once the process done, you will be alerted with a window message as show in Figure D.4.
6. Then, click on 'Update weight' button and wait for it to process. Once the process done, you will be alerted with a window message as show in Figure D.4.

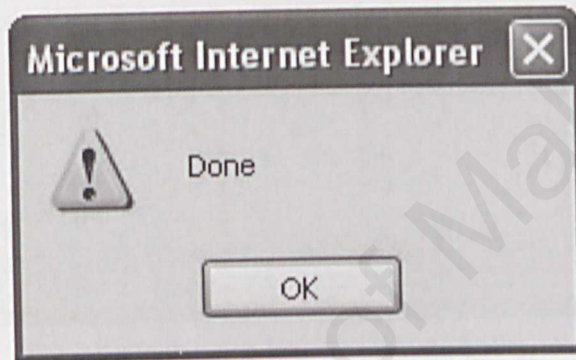


Figure D.4: window message shows the process is done completely

7. Then, you may close the page if you wish to. Or you may proceed to browse more files and repeat Step 2.

(ii) End-User: Search Page

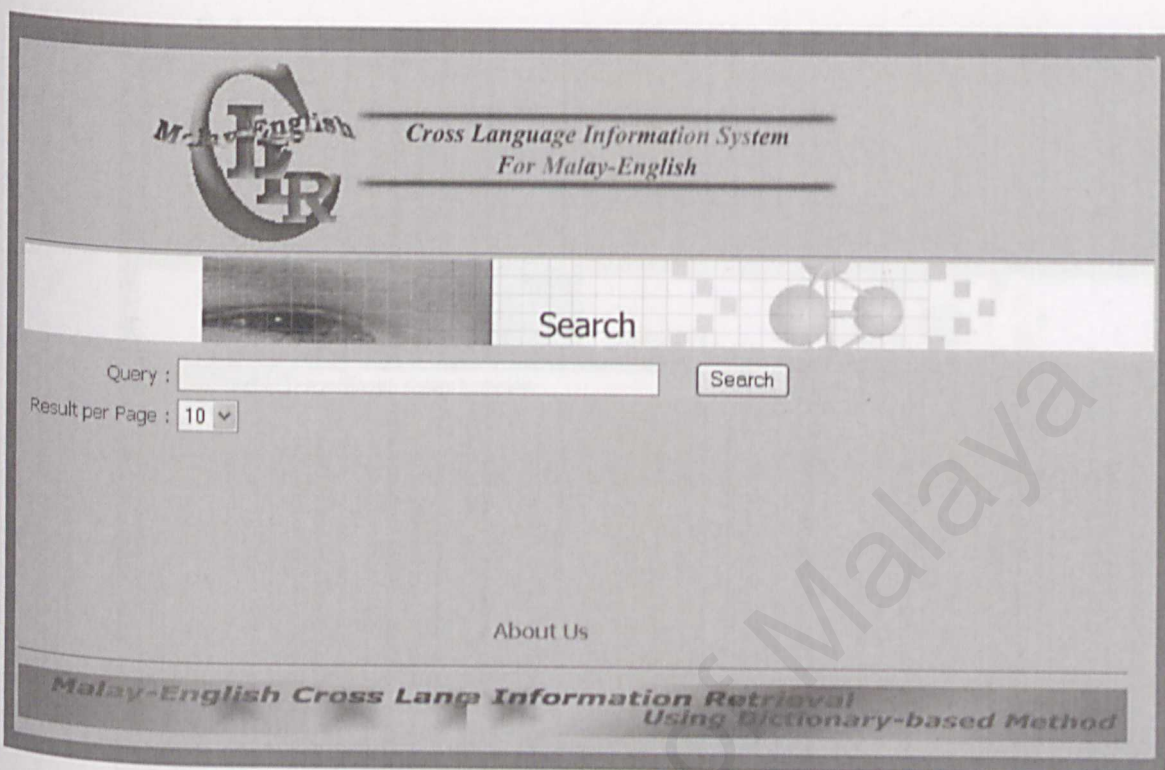


Figure D.5: Search Page

Instructions:

1. Type your information request or query in the input area. If not u will be prompted a window message as shown in Figure D.6.



Figure D.6: window message to show no query is provided by user

2. After entering your information request or query, click the 'Search' button.

3. After a few second, the result will be displayed as figure show in Figure D.7.

Malay-English Cross Language Information System
For Malay-English

Search

Query :

Result per Page :

Search Results Results 1 - 5 of 14 for kepentingan sistem komputer - 1.171114 sec

| 1 | 2 | 3 | <Next>

No	Title	Weight
1	A Medical-Operating Room Interaction System As part of this effort we are developing user interface technologies to facilitate the use of computer equipment in the OR. Our long-term goal is to provide automated support services	0.08653752
2	2 Inclination of scholars to major in information systems or computer science In light of the broadening scope of computing and because the feedback we received on our initial draft strongly encouraged us to move in this direction we have chosen to divide the CC2001 report into several volumes. This volume focuses specifically on computer science. To encompass...	0.06625128
3	A Multiple Cooperating Intelligent Agents Progress Report The use of robots for this research also requires the integration of complex machinery the development of large well engineered software systems and a number of emerging theoretical models - characteristics found throughout Computer Science. The Multiple Cooperating Agents Project (M...	0.04242036
4	A Multi-System Analysis of Document and Term Selection for Blind Feedback Experiments were conducted to explore the impact of combining various components of eight leading information retrieval systems . Each system demonstrated improved effectiveness with the use of blind feedback in which the results of a preliminary retrieval step were used to augment the ...	0.02679650
5	A Critical Review of the Notion of the Algorithm in Computer Science Computer science inherited its present conceptual foundations from a branch of pure mathematics that historically had been exploring the fundamental nature of mathematical computation since before the turn of the century. We will argue that the conceptual concerns of computer sc...	0.02431167

About Us

Malay-English Cross Language Information Retrieval
Using Dictionary-based Method

Figure D.7: Search Page

4. Click on the title of the document you wish to view.
5. If there is more than 1 page of results, click on the number linker or 'Prev' or 'Next' linker to navigate to that page.
6. If you wish to find more documents with another query, proceed by repeating Step 1 until Step 5.

University of Malaya

Appendix E: System Coding

```
<?
// retrieval/index.php

// System root document
$rootDoc = $_SERVER['SERVER_NAME'];
// Dir where text/doc stored
$textDir = $rootDoc."/texts";

// Load library
include './db/mysqlConnect.php'; // connection to MySQL db server
include './include/stemmer.inc';
include './include/dicTranslator.inc';
include './include/retrieval.inc';
include './include/publicFunction.inc';

if (isset($_REQUEST['searchSubmit'])) {
    $string = mysql_escape_string($_REQUEST['query']);
    $string = replace($string, " ", "");
}
?>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Search</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link href="./include/style.css" rel="stylesheet" type="text/css">
<script language="JavaScript">
function checkForm() {
    if (document.search.query.value.length < 1) {
        alert('Please type in your query. ');
        document.search.query.focus();
        return false;
    }
    else {
        document.search.submit();
    }
}
</script>
</head>
<body>
<table width="800" align="center" bgcolor="#C6D0DA">
<tr>
<td colspan="2">
<table width="530" bgcolor="#C6D0DA">
<tr>
<td width="734">
<div align="center">
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=6,0,29,0" width="600"
height="122">
<param name="movie" value="./images/title.swf">
<param name="quality" value="high">
<embed src="./images/title.swf" quality="high"
pluginspage="http://www.macromedia.com/go/getflashplayer" type="application/x-shockwave-flash" width="600"
height="122"></embed>
</object>
</div>
</td>
```

```

</tr>
<tr>
<td>
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=6,0,29,0" width="800"
height="57">
<param name="movie" value="../../../images/search.swf">
<param name="quality" value="high">
<embed src="../../../images/search.swf" quality="high"
pluginspage="http://www.macromedia.com/go/getflashplayer" type="application/x-shockwave-flash" width="800"
height="57"></embed>
</object>
</td>
</tr>
</table>
</td>
</tr>
<tr>
<td width="530" valign="top">
<table width="530" height="100%" class="statustext" align="center">
<form name="search" method="GET" action="<?=$_SERVER['PHP_SELF']?>">
<tr>
<td align="right">Query :</td>
<td><input type="text" name="query" size="50" value="<?=$string?>" style="statustext"></td>
<td align="left"><input type="submit" value="Search" onclick="checkForm()"></td>
</tr>
<tr>
<td align="right">Result per Page :</td>
<td colspan="2" align="left">
<?
$option = array("5", "10", "15");
if (isset($_REQUEST['totalShown']))
    $selected = $_REQUEST['totalShown'];
else
    $selected = '10';

HtmlPrintSelect($option, 'totalShown', $selected)?></td>
</tr>
</table>
<input type="hidden" value="submit" name="searchSubmit">
</form>
</td>
</tr>
</tr>
<td>
<?
if (isset($_REQUEST['searchSubmit'])) {
$result = array();
$rows = array();

$string = preg_replace('/[^a-zA-Z0-9- ]/', "", $string);

if (isset($_REQUEST['start']))
    $start = $_REQUEST['start'];
else
    $start = 0;

if (isset($_REQUEST['totalShown']))
    $totalShown = $_REQUEST['totalShown'];

```

```

$startTime = microtime(); // microtime before retrieving

$result = dicTranslator($string);

$text = "";
    for ($i=0; $i<sizeof($result); $i++){

        for ($j=0; $j<sizeof($result[$i]); $j++){
            $text .= $result[$i][$j]. " ";
        }
    }

$rows = retrieval($text);

// $respondTime is calculated by differentiate the 2 microtimes
$respondTime = microtime_diff($startTime, microtime());

```

```

$results = array();

// $results is an array that stored the results that going to display.
for($i=$start; $i < $start+$totalShowned && $rows[$i] != NULL; $i++){
    array_push($results, $rows[$i]);
}
$totalDoc = sizeof($rows);
$totalDisplayed = sizeof($results);

$begin = 0; // the number of first result retrieved of the page
$end = 0; // the number of last result retrieved of the page
// set value for $begin & $start if there number of result > 0
if ($totalDisplayed > 0){
    $begin = $start + 1;
    $end = $start + $totalDisplayed;
    $count = $begin;
}

```

```

<table width="100%" height="100%" class="spacing2" bgcolor="#C6D0DA">
<tr>
    <td colspan="3" bgcolor="#7892AD">
        <table width="100%">
            <tr>
                <td class="frametext" align="left" width="30%">Search Results</td>
                <td class="statustext" align="right" width="70%">
                    Results <b><?=$begin.> - "<?=$end.>" of <b><?=$totalDoc.></b>
                    for <?=$string.> - "<?=$respondTime.> sec
                </td>
            </tr>
        </table>
    </td>
</tr>
<tr>
    <td colspan="3" height="20"></td>
</tr>
<tr>
    <td colspan="3" align="center" class="frametext" id="link">
        <p><?
        $linker = pageLinker($totalDoc, $totalShowned, $_SERVER['PHP_SELF'], $start,
            "&query=$string&searchSubmit=submit", 'start', 'totalShowned');
        </p>
    </td>
</tr>

```

```

        echo $linker;
        ?></p>
    </td>
</tr>
<tr>
    <td colspan="3" height="20"></td>
</tr>
<tr class="topic">
    <td align="center" width="5%"><b>No</b></td>
    <td align="center" width="70%"><b>Title</b></td>
    <td align="center" width="25%"><b>Weight</b></td>
</tr>
<?
foreach ($results as $line) {
    $title = $line['title'];
    $weight = $line['weight'];
    $fulltext = $line['full_text'];
    $bolded = boldKey($fulltext, $keyWords);
    $bolded_text = "";
    foreach ($bolded as $bolded_word){

        $bolded_text .= $bolded_word;
        $bolded_text .= " ";
    }
    $pos = strpos($bolded_text, "<b>");
    $n = 0;

    // $n is total char between first bold_word and its previous
    // nearest foot stop.
    for($i=$pos-1; substr($bolded_text, $i, 1) != "." ; $i--){
        $n++;
    }

    if ($pos-$n > 0){
        $bolded_text = substr($bolded_text,$pos-$n,300);
    }

    else
        $bolded_text = substr($bolded_text,0,300);
    ?>
<tr>
    <td align="center" class="statustext"><?=$count++?></td>
    <td><table>
        <tr>
            <td id="topiclink"><a href="http://<?=$textDir."/"/>.$title.".txt"?>"
                target="_blank\" target="_new\"><?=$title?></a></td>
            </tr>
        </table>
    </td>
    <td align="center" class="statustext"><?=$weight?></td>
</tr>
<tr>
    <td></td>
    <td align="left" class="statustext"><?=$bolded_text?>...</td>
</tr>
<tr>
    <td colspan="2" height="30"></td>
    <?
    ?>
    }
    ?>
</tr>

```

```

<td colspan="3" align="center" class="frametext" id="link">
  <p class="statustext">Results Page:</p>
  <p><?
    echo $linker;
  ?></p>
</td>
</tr>
<?
  //endif (totaldoc > 0)
else {
  ?>
  <tr>
    <td colspan="3" height="20" class="topic" align="center">Result Not Found</td>
  </tr>
  <?
  }
?>
</table>
<?
}
?>

  </td>
</tr>
<tr>
  <td height="150"></td>
</tr>
<tr>
  <td colspan="4" align="center" id="topiclink"><a href="../aboutus.php">About Us </a></td>
</tr>
<tr>
  <td>
    
    
  </td>
</tr></table>
</body>
</html>

```

```

<?
// admin/index.php

// Load Libraries
include '../db/mysqlConnect.php'; // connection to MySQL db server
include '../include/stemmer.inc';
include '../include/indexing.inc';
include '../include/publicFunction.inc';

```

```

set_time_limit(0); // set time limit to infinity

```

```

if ($_REQUEST['action'] == 'updateIDF'){
    //calculate the value of inverted term doc and store the data into the database
    $result_update_i = countInvert();

```

```

    if ($result_update_i)
        echo "<script>alert('Done')</script>";
    else
        echo "<script>alert('Failed')</script>";
}

```

```

if ($_REQUEST['action'] == 'updateWeight'){
    //calculate the value of weight and store the data into the database
    $result_update_w = CountWeight();

```

```

    if ($result_update_w)
        echo "<script>alert('Done')</script>";
    else
        echo "<script>alert('Failed')</script>";
}

```

```

if (isset($_POST['thresholdSubmit'])){
    $newValue = $_POST['threshold'];

```

```

    $query = "UPDATE $tblname5 SET threshold = '$newValue'";

```

```

    $result = mysql_query($query);

```

```

    if ($result){
        $messageThres = "New threshold value is updated successfully! The new threshold value is <font
color=red>$newValue</font>";
    }

```

```

    else

```

```

        $messageThres = "New threshold is not updated successfullly!";
}

```

```

//When upload file button is clicked
if (isset($_POST['Upload'])){

```

```

    //Generate a random file id for the uploaded document
    $doc_id = randValue();

```

```

    //If no document browsed, the program exit
    if ($document=="none"){
        $message = "Problem: no file uploaded";
    }

```

```

    //If no the size of document is 0, the program exit
    if ($document_size==0){
        $message = "Problem: uploaded file is zero length";
    }

```

```

}

//If the type of document is not text file, the program exit
if ($document_type != "text/plain"){
    $message = "Problem: file is not plain text";
}

//If the document is failed to be uploaded, the program exit
if (!is_uploaded_file($document)){
    $message = "Problem: possible file upload attack";
}

//open and read the file
$fd = fopen ($document, "r");

//get the contents of the document
$content = @fread ($fd, filesize ($document));

//close the file
@fclose ($fd);

//Get the title of the document uploaded
$document_name = substr($document_name,0,-4);

if(!isExistInTable($document_name, "title", $tblname2)){
    //Insert the details of the document into the database
    $resultDoc = InsertDoc($doc_id, $document_name,$content);

    //Stem the contents
    $stemmed_content = stemming($content);

    //If the contents are stemmed and returned in array
    if (is_array($stemmed_content)) {

        //Display the stemmed contents in words instead of array
        foreach ( $stemmed_content as $word ){
            $string .= $word;
            $string .= " ";
        }
    }

    //Assign the text of the textareas to the variable $stemmed_text
    $stemmed_text = $string;

    //Convert the string of $stemmed_text to an array
    if ( !is_array($stemmed_text)){
        $stemmed_text = split("[ ,;\n\r\t]+", trim($stemmed_text));
    }

    //Sort the array from a-z
    sort($stemmed_text);

    //Get the number of words in the array
    $count = count($stemmed_text);

    //Count the same words in a document
    $count_sim = 1;

    for ($i=0; $i<sizeof($stemmed_text); $i++){

        //by default, the first word has another similar word
        $bInSim = true;
    }
}

```

```

//the current element of the array
$element = current($stemmed_text);

//compare the current element with the next element
//if they are similar, count the number of similar words
if(strpos($element, next($stemmed_text))!==0 && $bInSim){
    $bInSim = true;
    $count_sim++;
}
else {
    $bInSim = false;

    if($count_sim > 1)
        $finalCount = $count_sim;

```

```

    else
        $finalCount = 1;

```

```

    $count_sim = 1;
}

```

```

//calculate the frequency of similar terms and store the data into the database
if (!$bInSim){

```

```

    $term_freq = $finalCount/$count;
    $result = InsertData($element,$term_freq,"","",$doc_id);
}
}

```

```

echo "<SCRIPT LANGUAGE='JavaScript'">
    window.status = 'Document Indexs Stored Successfully!';
</SCRIPT>";

```

```

}
else {
    echo "<SCRIPT LANGUAGE='JavaScript'">
        window.status = 'This file is already in the database!';
    </SCRIPT>";
}
}

```

```

?>
<!-- Interface Design -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

```

```

<head>

```

```

    <title>Indexing</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    <link href="./include/style.css" rel="stylesheet" type="text/css">
    <!--

```

```

    .style1
    {

```

```

        font-family: Verdana;
        color: #000099;
        font-weight: bold;
        font-size: small;
    }

```

```

        .style9
        {
        font-size: x-small;
        color: #333366;
        }
        -->
    </style>
    <SCRIPT language=JavaScript src="../include/submit.js" type=text/javascript>
    </SCRIPT>
</head>
<body>
<table width="800" height="100%" align="center" bgcolor="#C6D0DA">
<tr>
<td>
<table width="700" bgcolor="#C1D0DF">
<tr>
<td width="734">
<div align="center">
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=6,0,29,0" width="600"
height="122">
<param name="movie" value="../images/title.swf">
<param name="quality" value="high">
<embed src="../images/title.swf" quality="high"
pluginspage="http://www.macromedia.com/go/getflashplayer" type="application/x-shockwave-flash" width="600"
height="122"></embed>
</object>
</div>
</td>
</tr>
<tr>
<td></td>
</tr>
</table>
</td>
</tr>
<tr>
<td>
<table id="indexing" width="100%" height="100%" align="center">
<form name = "upload" enctype="multipart/form-data" action="<?=$_SERVER['PHP_SELF']?>"
method="post">
<tr>
<td align="center"><font color="red"><?=$message?></font></td>
<td></td>
<td></td>
</tr>
<tr>
<td align="center" class="statustext">
<div>
<span class="style9">Upload new file:</span>
<input type="hidden" name="MAX_FILE_SIZE" value="1000">
<input name="document" type="file">
</div>
</td>
<td></td>
<td></td>
</tr>
<tr>
<td align="center"><a href="javascript:btnUpload()">
<input type="hidden" value="Upload this file" name="Upload"></a></td>
<td></td>
<td></td>
</tr>

```

```

</tr>
</form>
<tr>
    <td align="center" colspan="3">
        <input type="button" border="0" value="Update IDF"
onclick="location='<?=$_SERVER['PHP_SELF']?>?action=updateIDF'">
        <input type="button" border="0" value="Update Weight"
onclick="location='<?=$_SERVER['PHP_SELF']?>?action=updateWeight'">
    </td>
</tr>
<tr>
    <td colspan="3" align="center" class="statustext"><?=$messageThres?></td>
</tr>
<tr>
    <td colspan="3">
        <?
$threshold = getThreshold();
?>
        <form method="post" action="<?=$_SERVER['PHP_SELF']?>">
        <td colspan="3" align="center" class="statustext">Update threshold value:
            <input type="text" name="threshold" size="5" value="<?=$threshold?>">
            <br>
            <input type="submit" name="thresholdSubmit" value="Update"></td>
        </form>
    </tr>
<tr>
    <td colspan="3">
        <td colspan="3" align="center" id="topiclink">
            <a href=" ../aboutus.php">About Us </a>|
            <a href=" ../retrieval/">Malay-English CLIR</a></td>
        </tr>
    </table>
</td>
</tr>
<tr><td height="15"></td></tr>
</table>
</body>
</html>

```

```

<?
// include/translator.inc

/*****
* function name : ngram
* parameter : $input
* $input - Untranslatable word (string)
*
* Parameter $input is analyzed. The similarities of the input
* with the entries in dictionary are counted and the entry(s)
* with highest similarity will be return.
*
* return array nearest words set
*****/

function ngram ($input) {

    global $tblname1; // table name for dictionary

    // Query to MySQL
    $query = "SELECT * FROM $tblname1";
    $result = mysql_query($query);

    // Variable initialization
    $output = array();
    $temp = array();
    $word = array();
    $buf = array();
    $inputNgram = array();
    $maxSim = 0;

    // Ngram input word and put it as an array form
    for ($i=0; $i< strlen($input) - 1; $i++){
        array_push($inputNgram, substr($input, $i, 2));
    }

    // Evaluate similarity of the input word and words in dict
    while ($line = mysql_fetch_array($result)){
        $dicNgram = array();
        $dicNgram = explode(" ", $line['ngram']);

        /*
        * similarity = |inputNgram (intersection) dicNgram|
        * -----
        * |inputNgram (union) dicNgram|
        */
        $notSimilar = sizeof(array_diff($inputNgram, $dicNgram));
        $similar = sizeof($inputNgram) - $notSimilar;
        $divider = sizeof($dicNgram) + $notSimilar;
        $similarity = $similar/$divider;

        /*
        * This function only store the nearest word. This means the
        * buffer only stores the word(s) with highest similarity.
        */
        if ($similarity >= $maxSim){

            /*
            * There are possibility that the highest sim belong to
            * 2 or more words. In this case, word is pushed into
            * buffer without other operation.
            */

```

```

        */
        if ($similarity == $maxSim){
            array_push($buf, $line['enWord']);
        }
        /*
        * However, if there is higher similarity, previous stored word(s)
        * will be cleaned and then the higher sim word will be added.
        */

        elseif ($similarity > $maxSim){
            $buf = array();
            array_push($buf, $line['enWord']);
        }
        $maxSim = $similarity;
    }
}

```

```

/* Translated words normally consist of synonyms or word senses.
* These synonyms and word senses are separated by ';' and '@'.
*/

```

```

for ($i=0; $i<sizeof($buf); $i++){

    $temp = explode('@', $buf[$i]);

    foreach ($temp as $wordSense){
        if ($wordSense != ""){
            $word = explode(';', $wordSense);
            $output = array_merge($output, $word);
        }
    }
}

return $output;
}

```

```

/*****
* function name : ngram2
* parameter : $input
* $input - Untranslatable word (string)
*
* a same objective with ngram(). The only difference is
* that the total stored translated nearest word = $keep
* variable $keep can be modified to any number.
*
* return array nearest words set
*****/

```

```

function ngram2($input){

    global $tblname1; // table name for dictionary

    // query to MySQL
    $query = "SELECT * FROM $tblname1";
    $result = mysql_query($query);

    // Variable initialization
    $keep = 5; // Total number of result that will be kept
    $output = array();
    $temp = array();
}

```

```

$word = array();
$buf = array();
$inputNgram = array();
$lastSim = 0;

// Ngram input word and put it as an array form
for ($i=0; $i< strlen($input) - 1; $i++){
    array_push($inputNgram, substr($input, $i, 2));
}

// Evaluate similarity of the input word and words in dict
while ($line = mysql_fetch_array($result)){
    $dicNgram = array();
    $temp = array();
    $dicNgram = explode(" ", $line['ngram']);

    /*
    * similarity = |inputNgram (intersection) dicNgram|
    *               -----
    *               |inputNgram (union) dicNgram|
    */

    $notSimilar = sizeof(array_diff($inputNgram, $dicNgram));
    $similar = sizeof($inputNgram) - $notSimilar;
    $divider = sizeof($dicNgram) + $notSimilar;
    $similarity = $similar/$divider;

    /*
    * The below operation stores $keep word(s) into a buffer. The
    * elements (entries) are ordered by similarity (ascending).
    *
    * Note: $keep is no. of selected word(s) with $keep highest similarity.
    */

    /* Only operate if the current similarity > $lastSim
    * Note: $lastSim is the $keep-th entry's similarity.
    */

    if ($similarity > $lastSim){
        /*
        * The words stored in $buf are ordered by similarity (descending).
        * The method used here is to compare each stored word's similarity,
        * example below explains the process.
        * buf[6 5 3 2] <- [4]
        * 4 > 2   temp[2]         buf[6 5 3]
        * 4 > 3   temp[2 3]       buf[6 5]
        * 4 < 5   buf[6 5 4] += temp[2 3] => buf[6 5 4 3 2]
        */

        for ($i=sizeof($buf)-1; $i>=0; $i--){
            if ($similarity > $buf[$i]['sim']){
                array_push($temp, $buf[$i]);
                array_pop($buf);
            }
            else break;
        }

        if (sizeof($buf) < $keep){
            $line['sim'] = $similarity;
            array_push($buf, $line);
        }
    }
}

```

```

for ($i=sizeof($stemp)-1; $i >= 0 && sizeof($buf) < $keep; $i--){
    array_push ($buf, $stemp[$i]);
}

```

```

$bufSize = sizeof($buf);
if ($bufSize == $keep)
    $lastSim = $buf[$bufSize-1]['sim'];
}
}

```

```

/* Translated words normally consist of synonyms or word senses.
 * These synonyms and word senses are separated by ';' and '@'.
 */

```

```

for ($i=0; $i<sizeof($buf); $i++){

```

```

    $stemp = explode('@', $buf[$i]['enWord']);

```

```

    foreach ($stemp as $wordSense){
        if ($wordSense != ""){
            $word = explode(' ', $wordSense);
            $soutput = array_merge($soutput, $word);
        }
    }
}

```

```

return $soutput;
}

```

```

/*****
 * function name : Dictionary Lookup
 * parameter : $input
 * $input - Untranslatable word (string)
 *
 * Translate a Malay word e.g. komputer, or compound word e.g
 * salahguna, or phrase e.g. masa depan, sains komputer.
 * variable $keep can be modified to any number).
 *
 * return array translated words set
 *****/

```

```

function dictLookUp($input){

```

```

    global $tblname1;
    $buf = array();
    $stemp = array();
    $soutput = array();

```

```

    $query = "SELECT * FROM $tblname1 WHERE myWord = '$input'";
    $result = mysql_query($query);

```

```

    if (mysql_num_rows($result) > 0){

```

```

        /* Translated words normally consist of synonyms or word senses.
        * These synonyms and word senses are separated by ';' and '@'.
        */

```

```

        $entry = mysql_fetch_array($result);
        $buf = explode('@', $entry['enWord']);

```

```

        foreach ($buf as $wordSense){
            if ($wordSense != ""){

```

```

        for ($i=sizeof($stemp)-1; $i >= 0 && sizeof($buf) < $keep; $i--){
            array_push ($buf, $stemp[$i]);
        }

        $bufSize = sizeof($buf);
        if ($bufSize == $keep)
            $lastSim = $buf[$bufSize-1]['sim'];
    }
}

/* Translated words normally consist of synonyms or word senses.
 * These synonyms and word senses are separated by ';' and '@'.
 */

for ($i=0; $i<sizeof($buf); $i++){

    $stemp = explode('@', $buf[$i]['enWord']);

    foreach ($stemp as $wordSense){
        if ($wordSense != ""){
            $word = explode(';', $wordSense);
            $soutput = array_merge($soutput, $word);
        }
    }
}

return $soutput;
}

```

```

/*****
 * function name : Dictionary Lookup
 * parameter : $input
 * $input - Untranslatable word (string)
 *
 * Translate a Malay word e.g. komputer, or compound word e.g
 * salahguna, or phrase e.g. masa depan, sains komputer.
 * variable $keep can be motified to any number).
 *
 * return array translated words set
 *****/

```

```

function dictLookUp($input){

    global $tblname1;
    $buf = array();
    $stemp = array();
    $soutput = array();

    $query = "SELECT * FROM $tblname1 WHERE myWord = '$input'";
    $result = mysql_query($query);

    if (mysql_num_rows($result) > 0){

        /* Translated words normally consist of synonyms or word senses.
        * These synonyms and word senses are separated by ';' and '@'.
        */
        $entry = mysql_fetch_array($result);
        $buf = explode('@', $entry['enWord']);

        foreach ($buf as $wordSense){
            if ($wordSense != ""){

```

```

        $temp = explode(' ', $wordSense);
        $output = array_merge($output, $temp);
    }
}
return $output;
}

```

```

/*****
* function name : dicTranslator
* parameter : $string
* $string - user query (string)
*
* Processes in this function:
* a. Identify the language of the query
* b. Dictionary lookup
* c. Abbreviation Checking
* d. Ngram Handling (use ngram() (see function ngram()))
* e. Query structuring
*
* return 2 dimensional array structured English query
*****/

```

```

function dicTranslator ($string){

    // Variable initialization
    global $tblname1;
    global $tblname4;
    $output = array();
    $temp = array();
    $en = 0;
    $my = 0;

    // The string should be converted into array for better manipulation
    $input = explode(' ', $string);

    /* Mainly to remove extra white space between words. This is done
    * by removing empty element in $input
    */
    foreach ($input as $word){
        if ($word != NULL){
            array_push($temp, $word);
        }
    }
    $input = $temp;

    /* Process: Query language identification
    * Is MALAY if total of MALAY term found more than total of English term found
    * Is ENGLISH if total of English term found more than total of Malay term found
    *
    * Result:
    * If query language is MALAY, go to other dictionary lookup process
    * If query language is ENGLISH, return the original input string ($string)
    */
    foreach ($input as $word){
        $result = mysql_query("SELECT * FROM $tblname1 WHERE enWord LIKE '%$word%'");
        if (mysql_num_rows($result) > 0)
            ++$en;
    }

    foreach ($input as $word){

```

```

$result = mysql_query("SELECT * FROM $tblname1 WHERE myWord LIKE '%$word%'");
if (mysql_num_rows($result) > 0)
    ++$my;
}

if ($en > $my)
    $language = 'ENGLISH';

elseif ($my > $en)
    $language = 'MALAY';

else
    $language = 'MALAY'; // malay query is default language

if ($language == 'ENGLISH'){
    array_push($output, array("$string"));
    return $output;
}

elseif ($language == 'MALAY'){

// Do while $input is not empty
while (sizeof($input) > 0){

    $translated = array();
    $inputSize = sizeof($input); // original size of $input

    for ($i=0; $i<sizeof($input); $i++){
        $string = array2str($input, $i);

        /* Process: Dictionary Lookup ( See dictLookUp() )
        * Further Info:
        * If found translated word, put it to structure target language query
        * Else no result returned, go to abbreviation checking
        */
        $translated = dictLookUp($string);

        if (sizeof($translated) > 0){

            array_push($output, $translated);

            for ($j=0; $j<$inputSize-$i; $j++){
                array_pop($input);
            }
            break;
        }
    }

    else
    if ($i == sizeof($input)-1 && sizeof($translated) == 0){
        /*
        * Process: Abbreviation Checking
        * Condition: $input[] is last most word and untranslatable
        * Result:
        * If abbreviation is found, go to query construction
        * Else no result returned, go to Ngram handling
        */

        $query = "SELECT * FROM $tblname4 WHERE abbreviation = '$string'";
        $result = mysql_query($query);

        if (mysql_num_rows($result) > 0){

```

}

/*

*

```
Stemp = array();
```

3

```
array_pop($input);
```

}

}

3

```
$output = array_reverse($output);
```

```
*****
* function name : dicTranslator2
* parameter : $string
* $string - user query (string)
*
* Processes in this function:
* a. Identify the language of the query
* b. Dictionary lookup
* c. Abbreviation Checking
* d. Ngram Handling (use ngram() (see function ngram()))
* e. Query structuring
*
* return single dimensional array structured English query
*****
```

```
function dieTranslator2 ($string){
```

```
// Variable initialization
```

```
global $tblname1;
```

```
global $tblname1;  
global $tblname4;
```

```
Soutput = arrayC
```

```
Stemp = array();
```

```

$en = 0;
$my = 0;

// The string should be converted into array for better manipulation
$input = explode(' ', $string);

/* Mainly to remove extra white space between words. This is done
 * by removing empty element in $input
 */
foreach ($input as $word){
    if ($word != NULL){
        array_push($temp, $word);
    }
}
$input = $temp;

/* Process: Query language identification
 * Is MALAY if total of MALAY term found more than total of English term found
 * Is ENGLISH if total of English term found more than total of Malay term found
 *
 * Result:
 * If query language is MALAY, go to other dictionary lookup process
 * If query language is ENGLISH, return the original input string ($string)
 */
foreach ($input as $word){
    $result = mysql_query("SELECT * FROM $tblname1 WHERE enWord LIKE '%$word%'");
    if (mysql_num_rows($result) > 0)
        ++$en;
}

foreach ($input as $word){
    $result = mysql_query("SELECT * FROM $tblname1 WHERE myWord LIKE '%$word%'");
    if (mysql_num_rows($result) > 0)
        ++$my;
}

if ($en > $my)
    $language = 'ENGLISH';

elseif ($my > $en)
    $language = 'MALAY';

else
    $language = 'MALAY'; // malay query is default language

if ($language == 'ENGLISH'){
    array_push($output, $string);
    return $output;
}

elseif ($language == 'MALAY'){

// Do while $input is not empty
while (sizeof($input) > 0){
    $translated = array();
    $inputSize = sizeof($input); // original size of $input

    for ($i=0; $i<sizeof($input); $i++){
        $string = array2str($input, $i);

        /* Process: Dictionary Lookup ( See dictLookUp() )
        *

```

```

* Further Info:
* If found translated word, put it to structure target language query
* Else no result returned, go to abbreviation checking
*/

```

```

$translated = dictLookUp($string);

```

```

if (sizeof($translated) > 0){

```

```

    $output = array_merge($output, $translated);
    for ($j=0; $j<$inputSize-$i; $j++){
        array_pop($input);
    }
    break;
}

```

```

else

```

```

if ($i == sizeof($input)-1 && sizeof($translated) == 0){
    /*

```

```

        * Process: Abbreviation Checking
        * Condition: $input[] is last most word and untranslatable
        * Result:
        * If abbreviation is found, go to query construction
        * Else no result returned, go to Ngram handling
        */

```

```

        $query = "SELECT * FROM $tblname4 WHERE abbreviation = '$string'";
        $result = mysql_query($query);

```

```

        if (mysql_num_rows($result) > 0){

```

```

            $row = mysql_fetch_object($result);
            $full = explode(' ', $row->fullName);
            $fullName = array2str($full, 0);
            $fullName .= " $string";
            array_push($output, $fullName);
            array_pop($input);
        }

```

```

    else {
        /*

```

```

            * Process: Ngram Handling
            * Condition: $input[] is last most word and untranslatable
            * and not found in abbreviation list
            *
            * Find the nearest word(s) (see ngram()), and bring both
            * nearest word(s) and the untranslatable to query structuring
            */

```

```

            $ngram = ngram($string);
            $temp = array();

```

```

            for ($j=0; $j<sizeof($ngram); $j++){
                array_push($temp, $ngram[$j]);
            }
            array_push($temp, $string);
            $output = array_merge($output, $temp);
            array_pop($input);
        }
    }
}
}

```

```
$output = array_reverse($output);
```

```
return $output;
```

```
}
```

```
*****
```

```
* function name : dicTranslator3
```

```
* parameter : $string
```

```
* $string - user query (string)
```

```
* Processes in this function:
```

```
* a. Identify the language of the query
```

```
* b. Dictionary lookup
```

```
* c. Abbreviation Checking
```

```
* d. Ngram Handling (use ngram() (see function ngram()))
```

```
* e. Query structuring
```

```
* return 2 dimensional array structured English query
```

```
*****/
```

```
function dicTranslator3 ($string){
```

```
    // Variable initialization
```

```
    global $tblname1;
```

```
    global $tblname4;
```

```
    $output = array();
```

```
    $temp = array();
```

```
    $en = 0;
```

```
    $my = 0;
```

```
    // The string should be converted into array for better manipulation
```

```
    $input = explode(' ', $string);
```

```
    /* Mainly to remove extra white space between words. This is done  
    * by removing empty element in $input  
    */
```

```
    foreach ($input as $word){
```

```
        if ($word != NULL){
```

```
            array_push($temp, $word);
```

```
        }
```

```
    }
```

```
    $input = $temp;
```

```
    /* Process: Query language identification
```

```
    * Is MALAY if total of MALAY term found more than total of English term found
```

```
    * Is ENGLISH if total of English term found more than total of Malay term found
```

```
    *
```

```
    * Result:
```

```
    * If query language is MALAY, go to other dictionary lookup process
```

```
    * If query language is ENGLISH, return the original input string ($string)
```

```
    */
```

```
    foreach ($input as $word){
```

```
        $result = mysql_query("SELECT * FROM $tblname1 WHERE enWord LIKE '%$word%'");
```

```
        if (mysql_num_rows($result) > 0)
```

```
            ++$en;
```

```
    }
```

```
    foreach ($input as $word){
```

```
        $result = mysql_query("SELECT * FROM $tblname1 WHERE myWord LIKE '%$word%'");
```

```
        if (mysql_num_rows($result) > 0)
```

```
            ++$my;
```

```

}

if ($en > $my)
    $language = 'ENGLISH';

elseif ($my > $en)
    $language = 'MALAY';

else
    $language = 'MALAY'; // malay query is default language

if ($language == 'ENGLISH'){
    array_push($output, array("$string"));
    return $output;
}

elseif ($language == 'MALAY'){

// Do while $input is not empty
while (sizeof($input) > 0){

    $translated = array();
    $inputSize = sizeof($input); // original size of $input

    for ($i=0; $i<sizeof($input); $i++){
        $string = array2str($input, $i);

        /* Process: Dictionary Lookup ( See dictLookUp() )
        *
        * Further Info:
        * If found translated word, put it to structure target language query
        * Else no result returned, go to abbreviation checking
        */
        $translated = dictLookUp($string);

        if (sizeof($translated) > 0){

            array_push($output, $translated);

            for ($j=0; $j<$inputSize-$i; $j++){
                array_pop($input);
            }
            break;
        }
    }

    else
        if ($i == sizeof($input)-1 && sizeof($translated) == 0){
            /*
            * Process: Abbreviation Checking
            * Condition: $input[] is last most word and untranslatable
            * Result:
            * If abbreviation is found, go to query construction
            * Else no result returned, go to Ngram handling
            */

            $query = "SELECT * FROM $tblname4 WHERE abbreviation = '$string'";
            $result = mysql_query($query);

            if (mysql_num_rows($result) > 0){

                $row = mysql_fetch_object($result);
                $full = explode(' ', $row->fullName);
            }
        }
    }
}

```

```

        $fullName = array2str($full, 0);
        $fullName .= " $string";
        array_push($output, array("$fullName"));
        array_pop($input);
    }

    else {
        /*
        * Process: Ngram Handling
        * Condition: $input[] is last most word and untranslatable
        *             and not found in abbreviation list
        *
        * Find the nearest word(s) (see ngram2()), and bring both
        * nearest word(s) and the untranslatable to query structuring
        */

        $ngram = ngram2($string);
        $temp = array();

        for ($j=0; $j<sizeof($ngram); $j++){
            //$a = dictLookUp($ngram[$j]);
            array_push($temp, $ngram[$j]);
        }
        array_push($temp, $string);
        array_push($output, $temp);
        array_pop($input);
    }
}

}

$output = array_reverse($output);

return $output;
}

```

```

/*****
* function name : dicTranslator4
* parameter : $string
* $string - user query (string)
*
* Processes in this function:
* a. Identify the language of the query
* b. Dictionary lookup
* c. Abbreviation Checking
* d. Ngram Handling (use ngram2() (see function ngram2()))
* e. Query structuring
*
* return single dimensional array structured English query
*****/

```

```

function dicTranslator4 ($string){

```

```

    // Variable initialization
    global $tblname1;
    global $tblname4;
    $output = array();
    $temp = array();
    $sen = 0;
    $my = 0;

```

```
// The string should be converted into array for better manipulation
$input = explode(' ', $string);
```

```
/* Mainly to remove extra white space between words. This is done
 * by removing empty element in $input
 */
```

```
foreach ($input as $word){
    if ($word != NULL){
        array_push($temp, $word);
    }
}
$input = $temp;
```

```
/* Process: Query language identification
```

```
 * Is MALAY if total of MALAY term found more than total of English term found
 * Is ENGLISH if total of English term found more than total of Malay term found
 *
```

```
 * Result:
```

```
 * If query language is MALAY, go to other dictionary lookup process
 * If query language is ENGLISH, return the original input string ($string)
 */
```

```
foreach ($input as $word){
    $result = mysql_query("SELECT * FROM $tblname1 WHERE enWord LIKE '%$word%'");
    if (mysql_num_rows($result) > 0)
        ++$en;
```

```
}
foreach ($input as $word){
    $result = mysql_query("SELECT * FROM $tblname1 WHERE myWord LIKE '%$word%'");
    if (mysql_num_rows($result) > 0)
        ++$my;
```

```
}
if ($en > $my)
    $language = 'ENGLISH';
```

```
elseif ($my > $en)
    $language = 'MALAY';
```

```
else
    $language = 'MALAY'; // malay query is default language
```

```
if ($language == 'ENGLISH'){
    array_push($output, $string);
    return $output;
}
```

```
elseif ($language == 'MALAY'){
```

```
    // Do while $input is not empty
```

```
    while (sizeof($input) > 0){
        $translated = array();
        $inputSize = sizeof($input); // original size of $input
```

```
        for ($i=0; $i<sizeof($input); $i++){
            $string = array2str($input, $i);
```

```
            /* Process: Dictionary Lookup ( See dictLookUp() )
```

```
            *
```

```
            * Further Info:
```

```
            * If found translated word, put it to structure target language query
            * Else no result returned, go to abbreviation checking
            */
```

```

$translated = dictLookUp($string);

if (sizeof($translated) > 0){

    $output = array_merge($output, $translated);
    for ($j=0; $j<$inputSize-$i; $j++){
        array_pop($input);
    }
    break;
}
else
if ($i == sizeof($input)-1 && sizeof($translated) == 0){
    /*
    * Process: Abbreviation Checking
    * Condition: $input[] is last most word and untranslatable
    * Result:
    * If abbreviation is found, go to query construction
    * Else no result returned, go to Ngram handling
    */

    $query = "SELECT * FROM $tblname4 WHERE abbreviation = '$string'";
    $result = mysql_query($query);

    if (mysql_num_rows($result) > 0){

        $row = mysql_fetch_object($result);
        $full = explode(' ', $row->fullName);
        $fullName = array2str($full, 0);
        $fullName .= " $string";
        array_push($output, $fullName);
        array_pop($input);
    }
    else {
        /*
        * Process: Ngram Handling
        * Condition: $input[] is last most word and untranslatable
        * and not found in abbreviation list
        *
        * Find the nearest word(s) (see ngram2()), and bring both
        * nearest word(s) and the untranslatable to query structuring
        */
        $ngram = ngram2($string);
        $temp = array();

        for ($j=0; $j<sizeof($ngram); $j++){
            array_push($temp, $ngram[$j]);
        }
        array_push($temp, $string);
        $output = array_merge($output, $temp);
        array_pop($input);
    }
}

}

}

$output = array_reverse($output);

return $output;
}

```

```

<?
// include/publicFunction.inc

// Get threshold value from database
function getThreshold(){
    global $tblname5;

    $result = mysql_query("SELECT * FROM $tblname5");
    $row = mysql_fetch_object($result);

    return $row->threshold;
}

// Return stemmed word
function stemming($input)
{
    $stem = new Stemmer();
    $stemmed = $stem->stem_list($input);

    return $stemmed;
}

// Bold word in text if word == keyword
function boldKey($temptext, $keyWords)
{
    $temptext = split("[,;\n\r\t]", trim($temptext));
    $keyWords = split("[,;\n\r\t]", trim($keyWords));

    for($i=0;$i<sizeof($keyWords);$i++)
    {
        for($j=0;$j<sizeof($temptext);$j++)
        {
            $tempText = strtolower($temptext[$j]);
            $tempKeyword = strtolower($keyWords[$i]);
            if (strpos($tempText, $tempKeyword) &&
                substr($tempText,0,2)==substr($tempKeyword,0,2))
            {
                $temptext[$j] = "<b>$temptext[$j]</b>";
            }
            else
                $temptext[$j] = $temptext[$j];
        }
    }

    return $temptext;
}

// Combine all elements in $array to become a string
function array2str($array, $n){
    $output = "";
    while ($n < sizeof($array)){
        $output .= $array[$n++].',';
    }

    return $output;
}

// Reverse the sequence of words in a sentence (string)

```

```

function reverseStr($string){
    $temp = explode(' ', $string);
    $temp = array_reverse($temp);

    return array2str($temp, 0);
}

// Generate a random value
function randValue()
{
    $Date = date("Ymd");
    $randValue = (Int)((9000000 - 100000 + 1) * rand() + $Date + 10000000);
    if ($randValue < 0)
        $randValue = (Int)(-1 * $randValue);

    return $randValue;
}

```

```

// Build a drop down list
function HtmlPrintSelect($options, $selName, $selected){

    echo '<select name="'. $selName. '">';

    foreach($options as $val) {

        echo '<option value="'. $val. '"';

        if($val == $selected)

            echo ' selected';

        echo '>'. $val. '</option>';

    }

    echo '</select>';
}

```

```

// All $a (any char or punctuation mark) in string $input are replaced by $b
function replace($input, $a, $b){
    $handle = explode($a, $input);
    $output = "";
    for ($i=0; $i < count($handle); $i++) {
        if ($i < count($handle)-1)
            $output .= $handle[$i].$b;

        if ($i == count($handle)-1)
            $output .= $handle[$i];
    }
    return $output;
}

```

```

// Calculate time between microtime a & b
function microtime_diff($a, $b) {
    list($a_dec, $a_sec) = explode(" ", $a);
    list($b_dec, $b_sec) = explode(" ", $b);
    return $b_sec - $a_sec + $b_dec - $a_dec;
}

```

```

// Check a single data's existence in one table

```

```
function isExistInTable ($data, $field, $table){
```

```
    $sql = "SELECT * FROM $table WHERE $field = '$data'";
```

```
    $sql_result = mysql_query($sql);
```

```
    if (mysql_num_rows($sql_result) > 0) // if the data found in the table
        return true;
```

```
    else
        return false;
```

```
}
```

```
// Function that provide page linker
```

```
function pageLinker($totalResult, $totalPerPage, $targetURL,
    $start, $requestString, $startStr='start', $perPageStr='perpage'){
```

```
    $HTMLcode = "";
```

```
    $resultedHTMLcode = "";
```

```
    for ($i=0, $j=1; $i < $totalResult; $i += $totalPerPage, $j++){
```

```
        if ($i == $start){
```

```
            $HTMLcode .= "| $j ";
```

```
            $currentPage = $j;
```

```
            $previous = "";
```

```
            $next = "";
```

```
            // do not do this at first result page
```

```
            if ($i-$totalPerPage >= 0){
```

```
                $previous .= "<a href='\".$targetURL;
```

```
                $previous .= "?$startStr=\".($i-
```

```
            // do not do this at last result page
```

```
            if ($i+$totalPerPage < $totalResult){
```

```
                $next .= "<a href='\".$targetURL;
```

```
                $next .=
```

```
                "$startStr=\".($i+$totalPerPage).\".$perPageStr=$totalPerPage\".$requestString\"> &lt;Next&gt; </a>";
```

```
            }
```

```
        }
```

```
        else {
```

```
            $HTMLcode .= "| <a href='\".$targetURL;
```

```
            $HTMLcode .= "?$startStr=$i&$perPageStr=$totalPerPage\".$requestString\"> $j </a>";
```

```
        }
```

```
        // last page
```

```
        $lastPage = $j;
```

```
    }
```

```
    $HTMLcode .= "|";
```

```
    $temp = explode("|", $HTMLcode);
```

```
    $front = array();
```

```
    $back = array();
```

```
    $temp = array_filter($temp);
```

```
    $middleIndex = array_search($currentPage, $temp);
```

```
    // get 5 backward components
```

```
    for($i=1, $j=3; $j >= 1 && $middleIndex-$i > 0; $i++, $j--){
```

```
        array_push($front, $temp[$middleIndex-$i]);
```

```
    }
```

```
    // get 5 forward components
```

```
for($x=1, $y=$j+3; $y >= 1 && $middleIndex+$x <= $lastPage; $x++, $y--){  
    array_push($back, $temp[$middleIndex+$x]);  
}
```

```
if ($y > 0){  
    $front = array();  
    for($i=1, $j=$y+3; $j >= 1 && $middleIndex-$i > 0; $i++, $j--){  
        array_push($front, $temp[$middleIndex-$i]);  
    }  
}
```

```
$front = array_reverse($front);  
array_push($front, $middleIndex);
```

```
$temp = array_merge($front, $back);
```

```
foreach ($temp as $link){  
    $resultedHTMLcode .= "| $link ";  
}
```

```
$resultedHTMLcode .= " |";
```

```
$resultedHTMLcode = $previous.$resultedHTMLcode.$next;
```

```
return $resultedHTMLcode;
```

```

<?
//include/stemmer.inc

/*****
 *
 * class.stemmer.inc
 *
 *****/

Implementation of the Porter Stemming Algorithm

*****/

/*
 * Takes a word, or list of words, and reduces them to their English stems.
 *
 * There is a deviation in the way compound words are stemmed, such as
 * hyphenated words and words starting with certain prefixes.
 */
class Stemmer
{
    /**
     * function name : stem
     * Takes a word as parameter and returns it reduced to its stem.
     *
     * Non-alphanumerics and hyphens are removed, and if the word is more than
     * three characters in length, it will be stemmed according to the five-step
     * Porter stemming algorithm.
     *
     * return string Stemmed word
     */
    function stem( $word ) {

        if ( empty($word) ) {
            return false;
        }

        $result = "";

        //convert all the characters into lower cases
        $word = strtolower($word);

        // Strip punctuation, e.g:sister's, pupils'
        if ( substr($word, -2) == "'s" || substr($word, -2) == "'s'" ) {
            $word = substr($word, 0, -2);
        }

        //Lexical Analysis
        //Replace the characters other than alphabets
        //and number to nothing

        $word = preg_replace('/[^a-z0-9-]/', "", $word);

        //Stopwords Removal
        //Read the stop_word_list file
        $filename = "../include/stop_word_list.txt";
    }
}

```

```

$fp = fopen($filename, "r");

while (!feof($fp)) {
    $stop_words = $stop_word . fread($fp, 10000);
}

//split the list of stopwords into an array
$stop_words = split("[ ,\n\r\t]+", trim($stop_words));

//Compare each stopword with the word
foreach ($stop_words as $stop_word) {

    if (strcmp($word, $stop_word) == 0) {
        $bInStopWord = true;
    }
}

//if the word is not a stop word, the word will go through further
//Porter Stemming Algorithm

if ($bInStopWord != true) {

    if ( strlen($word) > 3 ) {

        $word = $this->step_1($word);
        $word = $this->step_2($word);
        $word = $this->step_3($word);
        $word = $this->step_4($word);
        $word = $this->step_5($word);

    }

    //Stemmed word
    $result = $word;

    return $result;
}
}

```

```

/*
 * Function Name : stem_list
 * Takes a list of words as parameter and returns them reduced to their stems.
 *
 * $words can be either a string or an array. If it is a string, it will
 * be split into separate words on whitespace, commas, or semicolons. If
 * an array, it assumes one word per element.
 *
 * return array List of word stems
 */

```

```

function stem_list( $words ) {

    //if the parameter received is empty, return false
    if ( empty($words) ) {
        return false;
    }

    //define an array to store results of stemming
    $results = array();

```

```
//if the parameter is not in an array form,
//convert it into an array

if ( !is_array($words) ) {
    $words = split("[\n\r\t]+", trim($words));
}

```

```
//stem the words 1 by 1
foreach ( $words as $word ) {
    if ( $result = $this->stem($word) ) {
        $results[] = $result;
    }
}

```

```
return $results;
}

```

```
/*
* function Name: step_1
* Performs the functions of steps 1a, 1b and 1c of the Porter Stemming Algorithm.
*
* 1(a) : If the word is in plural form, it is reduced to singular form.
* e.g :
* SSES -> SS      caresses -> caress
* IES -> I        ponies -> poni    ties -> ti
* SS -> SS        caress -> caress
* S ->            cats -> cat
*
* 1(b) : any -ed or -ing endings are removed as appropriate, and finally,
* words ending in "y" with a vowel in the stem have the "y" changed to "i".
* (m>0) EED -> EE    feed -> feed
* (*v*) ED ->        plastered -> plaster
*                      bled -> bled
* (*v*) ING ->       motoring -> motor
*                      sing -> sing
*
* 1(c) (*v*) Y -> I
* e.g :    happy -> happi
*          sky -> sky
* return string Reduced word
*/

```

```
function step_1( $word ) {
    //1(a)
    if ( substr($word, -1) == 's' ) {
        if ( substr($word, -4) == 'sses' ) {
            $word = substr($word, 0, -2);
        }

        elseif ( substr($word, -3) == 'ies' ) {
            $word = substr($word, 0, -2);
        }

        // If second-to-last character is not "s"
        //e.g : words -> word
        elseif ( substr($word, -2, 1) != 's' ) {
            $word = substr($word, 0, -1);
        }
    }

    //1(b)
    if ( substr($word, -3) == 'eed' && $this->count_vc(substr($word, -3)) > 0 ) {

```

```

// Convert -eed to -ee
$word = substr($word, 0, -1);
}
else {
    if ( preg_match('/[aeiou][^aeiou]*(ed|ing)$/', $word) ) {
        // vowel in stem
        // Strip -ed or -ing
        if ( substr($word, -2) == 'ed' ) {
            $word = substr($word, 0, -2);
        }

        else {
            $word = substr($word, 0, -3);
        }

        //If the second or third of the rules in Step 1b is
        //successful, the following is done:

        if ( substr($word, -2) == 'at' || substr($word, -2) == 'bl' || substr($word, -2) == 'iz' ) {
            $word .= 'e';
        }

        else {

            $last_char = substr($word, -1, 1);
            $next_to_last = substr($word, -2, 1);

            // Strip ending double consonants to single, unless "l", "s" or "z"
            //e.g : stripp  -> strip
            if ( $this->is_consonant($word, -1) && $last_char == $next_to_last &&
                $last_char != 'l' && $last_char != 's' && $last_char != 'z' )
            {
                $word = substr($word, 0, -1);
            }

            else {
                // If VC, and cvc (but not w,x,y)
                if ( $this->count_vc($word) == 1 && $this->o($word) ) {
                    $word .= 'e';
                }
            }
        }
    }
}
//1(c)
// Turn y into i when another vowel in stem
if ( preg_match('/[aeiou][^aeiou]*y$/', $word) ) { // vowel in stem
    $word = substr($word, 0, -1) . 'i';
}
return $word;
}

```

- /*
- * Function Name : step_2
 - * Performs the function of step 2 of the Porter Stemming Algorithm.
 - * Step 2 maps double suffixes to single ones when the second-to-last character matches the given letters.
 - * e.g :
 - * (m>0) ATIONAL -> ATE relational -> relate
 - * (m>0) TIONAL -> TION conditional -> condition

```

* (m>0) ENCI -> ENCE      valenci -> valence
* (m>0) ANCI -> ANCE      hesitanci -> hesitance
* (m>0) IZER -> IZE       digitizer -> digitize
*
* return string Reduced word
*/

```

```

function step_2( $word )
{

```

```

    switch ( substr($word, -2, 1) ) {

```

```

        case 'a':

```

```

            if ( $this->replace($word, 'ational', 'ate', 0) ) {
                return $word;
            }

```

```

            if ( $this->replace($word, 'tional', 'tion', 0) ) {

```

```

                return $word;
            }

```

```

            break;

```

```

        case 'c':

```

```

            if ( $this->replace($word, 'enci', 'ence', 0) ) {
                return $word;
            }

```

```

            if ( $this->replace($word, 'anci', 'ance', 0) ) {
                return $word;
            }

```

```

            break;

```

```

        case 'e':

```

```

            if ( $this->replace($word, 'izer', 'ize', 0) ) {
                return $word;
            }

```

```

            if ( $this->replace($word, 'iser', 'ize', 0) ) {
                return $word;
            }

```

```

            break;

```

```

        case 'l':

```

```

            if ( $this->replace($word, 'bli', 'ble', 0) ) {
                return $word;
            }

```

```

            if ( $this->replace($word, 'alli', 'al', 0) ) {
                return $word;
            }

```

```

            if ( $this->replace($word, 'entli', 'ent', 0) ) {
                return $word;
            }

```

```

            if ( $this->replace($word, 'eli', 'e', 0) ) {
                return $word;
            }

```

```

            if ( $this->replace($word, 'ousli', 'ous', 0) ) {
                return $word;
            }

```

```

            break;

```

```

        case 'o':

```

```

            if ( $this->replace($word, 'ization', 'ize', 0) ) {
                return $word;
            }

```

```

            if ( $this->replace($word, 'isation', 'ize', 0) ) {
                return $word;
            }

```

```

    }
    if ( $this->replace($word, 'ation', 'ate', 0) ) {
        return $word;
    }
    if ( $this->replace($word, 'ator', 'ate', 0) ) {
        return $word;
    }
    break;

case 's':
    if ( $this->replace($word, 'alism', 'al', 0) ) {
        return $word;
    }
    if ( $this->replace($word, 'iveness', 'ive', 0) ) {
        return $word;
    }
    if ( $this->replace($word, 'fulness', 'ful', 0) ) {
        return $word;
    }
    if ( $this->replace($word, 'ousness', 'ous', 0) ) {
        return $word;
    }
    break;

case 't':
    if ( $this->replace($word, 'aliti', 'al', 0) ) {
        return $word;
    }
    if ( $this->replace($word, 'iviti', 'ive', 0) ) {
        return $word;
    }
    if ( $this->replace($word, 'biliti', 'ble', 0) ) {
        return $word;
    }
    break;

case 'g':
    if ( $this->replace($word, 'logi', 'log', 0) ) {
        return $word;
    }
    break;
}
return $word;
}

```

```

/*
* Function Name : step_3
* Performs the function of step 3 of the Porter Stemming Algorithm.
* Step 3 works in a similar strategy to step 2, though checking the
* last character.
* e.g :
* (m>0) ICATE -> IC      triplicate -> triplic
* (m>0) ATIVE ->          formative -> form
* (m>0) ALIZE -> AL      formalize -> formal
* (m>0) ICITI -> IC      electriciti -> electric
* return string Reduced word
*/
function step_3( $word )
{
    switch ( substr($word, -1) ) {

```

```

case 'e':
    if ( $this->replace($word, 'icate', 'ic', 0) ) {
        return $word;
    }
    if ( $this->replace($word, 'ative', '', 0) ) {
        return $word;
    }
    if ( $this->replace($word, 'alize', 'al', 0) )
        return $word;
    }
    break;

case 'i':
    if ( $this->replace($word, 'iciti', 'ic', 0) ) {
        return $word;
    }
    break;

case 'l':
    if ( $this->replace($word, 'ical', 'ic', 0) ) {
        return $word;
    }
    if ( $this->replace($word, 'ful', '', 0) ) {
        return $word;
    }
    break;

case 's':
    if ( $this->replace($word, 'ness', '', 0) ) {
        return $word;
    }
    break;
}
return $word;
}

/**
 * Function Name : step_4
 * Performs the function of step 4 of the
 * Porter Stemming Algorithm.
 *
 * Step 4 works similarly to steps 3 and 2, above,
 * though it removes the endings in the context of VCVC
 * e.g:
 * (m>1) AL -> revival -> reviv
 * (m>1) ANCE -> allowance -> allow
 * (m>1) ENCE -> inference -> infer
 * (m>1) ER -> airliner -> airlin
 *
 * @return string Reduced word
 */
function step_4( $word )
{
    switch ( substr($word, -2, 1) ) {
        case 'a':
            if ( $this->replace($word, 'al', '', 1) ) {
                return $word;
            }
            break;

        case 'c':
            if ( $this->replace($word, 'ance', '', 1) ) {

```

```

        return $word;
    }
    if ( $this->replace($word, 'ence', '', 1) ) {
        return $word;
    }
    break;

case 'e':
    if ( $this->replace($word, 'er', '', 1) ) {
        return $word;
    }
    break;

case 'i':
    if ( $this->replace($word, 'ie', '', 1) ) {
        return $word;
    }
    break;

case 'l':
    if ( $this->replace($word, 'able', '', 1) ) {
        return $word;
    }
    if ( $this->replace($word, 'ible', '', 1) ) {
        return $word;
    }
    break;

case 'n':
    if ( $this->replace($word, 'ant', '', 1) ) {
        return $word;
    }
    if ( $this->replace($word, 'ement', '', 1) ) {
        return $word;
    }
    if ( $this->replace($word, 'ment', '', 1) ) {
        return $word;
    }
    if ( $this->replace($word, 'ent', '', 1) ) {
        return $word;
    }
    break;

case 'o':
    // special cases
    if ( substr($word, -4) == 'sion' || substr($word, -4) == 'tion' ) {
        if ( $this->replace($word, 'ion', '', 1) ) {
            return $word;
        }
    }
    if ( $this->replace($word, 'ou', '', 1) ) {
        return $word;
    }
    break;

case 's':
    if ( $this->replace($word, 'ism', '', 1) ) {
        return $word;
    }
    break;

```

```

case 't':
    if ( $this->replace($word, 'ate', "", 1) ) {
        return $word;
    }
    if ( $this->replace($word, 'iti', "", 1) ) {
        return $word;
    }
    break;

```

```

case 'u':
    if ( $this->replace($word, 'ous', "", 1) ) {
        return $word;
    }
    break;

```

```

case 'v':
    if ( $this->replace($word, 'ive', "", 1) ) {
        return $word;
    }
    break;

```

```

case 'z':
    if ( $this->replace($word, 'ize', "", 1) ) {
        return $word;
    }
    break;

```

```

}
return $word;
}

```

```

**
*Function Name : step_5
* Performs the function of step 5(a) and 5(b) of the
* Porter Stemming Algorithm.
*
* Step 5 removes a final "-e" and changes "-ll" to "-l"
* in the context of VCVC
*
* e.g :
* Step 5a
* (m>1) E -> probate -> probat
* rate -> rate
* (m=1 and not *o) E -> cease -> ceas
* Step 5b
* (m > 1 and *d and *L) -> single letter controll -> control
* roll -> roll
* return string Reduced word
*/

```

```

function step_5( $word ) {
    if ( substr($word, -1) == 'e' ) {
        $short = substr($word, 0, -1);
        // Only remove in vcvc context...
        if ( $this->count_vc($short) > 1 ) {
            $word = $short;
        }
        elseif ( $this->count_vc($short) == 1 && !$this->o($short) ) {
            $word = $short;
        }
    }
}

```

```

if ( substr($word, -2) == 'll' ) {
    // Only remove in vcvc context...
    if ( $this->count_vc($word) > 1 ) {
        $word = substr($word, 0, -1);
    }
}
return $word;
}

```

```

/*
 * Function Name : is_consonant
 * Takes a word and a number as parameters
 * Checks whether the specified letter is a consonant at
 * the specific position.
 */

```

```

* Handy check adapted from the ANSI C program.
* Regular vowels always return
* FALSE, while "y" is a special case:
* if the preceding character is a vowel,
* "y" is a consonant, otherwise it's a vowel.

```

```

* return boolean
*/

```

```

function is_consonant( $word, $pos ) {

```

```

    // Sanity checking the position of characters

```

```

    if ( abs($pos) > strlen($word) ) {

```

```

        if ( $pos < 0 ) {
            // Points "too far back" in the string. Set it to beginning.
            $pos = 0;

```

```

        }
        else {
            // Points "too far forward." Set it to end.
            $pos = -1;

```

```

        }
    }
    $char = substr($word, $pos, 1);

```

```

    switch ( $char ) {

```

```

        case 'a':

```

```

        case 'e':

```

```

        case 'i':

```

```

        case 'o':

```

```

        case 'u':

```

```

            return false;

```

```

        case 'y':

```

```

            if ( $pos == 0 || strlen($word) == -$pos ) {

```

```

                // Check second letter of word.

```

```

                return !($this->is_consonant($word, 1));

```

```

            }

```

```

            else {

```

```

                return !($this->is_consonant($word, $pos - 1));

```

```

            }

```

```

        default:

```

```

            return true;

```

```

        }
    }
}

```

```

/*
 * Function Name : count_vc

```

```

 * Counts (measures) the number of vowel-consonant occurrences.
 */

```

```

* This handy function counts the number of occurrences
*   of vowels (1 or more) followed by consonants (1 or more)
* A legitimate C combination counts as 1
* (ie. VCVC = 2, VCVCVC = 3, etc.).
*
* return integer
*/

```

```

function count_vc( $word ) {
    $m = 0;

    $length = strlen($word);

    $prev_c = false;

    for ( $i = 0; $i < $length; $i++ ) {

        $is_c = $this->is_consonant($word, $i);
        if ( $is_c ) {

            if ( $m > 0 && !$prev_c ) {
                $m += 0.5;
            }
        }
        else {
            if ( $prev_c || $m == 0 ) {
                $m += 0.5;
            }
        }

        $prev_c = $is_c;
    }
    $m = floor($m);

    return $m;
}

```

```

/*
* function Name : _o
* Checks for a specific cvc condition.
*
* It looks the last three characters of the word ending as
* consonant-vowel-consonant, with the final consonant NOT
* being one of "w", "x" or "y".
*
* return boolean
*/

```

```

function _o( $word ) {

    $last_char = substr($word, -1);

    if ( $last_char == 'w' || $last_char == 'x' || $last_char == 'y' ) {
        return false;
    }

    if ( strlen($word) >= 3 ) {

        if ( $this->is_consonant($word, -1) && !$this->is_consonant($word, -2)
            && $this->is_consonant($word, -3) ) {
            return true;
        }
    }
}

```

```

return false;
}

/*
 * Replaces suffix, if found and word measure is a minimum count.
 *
 * Takes 4 parameters
 * return boolean
 */
function replace( &$word, $suffix, $replace, $m = 0 )
{
    $sl = strlen($suffix);

    if ( substr($word, -$sl) == $suffix ) {
        $short = substr_replace($word, "", -$sl);

        if ( $this->count_vc($short) > $m ) {
            $word = $short . $replace;
        }

        return true;
    }
    return false;
}

?>

```

<?

//include/retrieval.inc

```
*****
* function name :retrieval *
* This function is retrieve or search the relevant documents by using the *
* translated query. *
* *
* It takes an argument and returns an array that stores all the relevant documents. *
* *
*****/
```

function retrieval(\$string) {

global \$textDir;
global \$link;
global \$keyWords;
global \$tblname5;

\$result = mysql_query("SELECT * FROM \$tblname5");
\$row = mysql_fetch_object(\$result);

\$threshold = \$row->threshold;

//define an array to store the results
\$rows = array();
\$temp = array();
\$weightSort = array();

//preprocess the translated query
\$translated = stemming(\$string);

if (sizeof(\$translated) > 0){

// Construct the SQL LIKE statement
foreach (\$translated as \$word) {

\$keyWords .= \$word;
\$keyWords .= " ";

\$pre_query .= "i.index_term LIKE ";
\$pre_query .= "'\$word%'";
\$pre_query .= " OR ";

}

// Cut off the " OR" at the end of the pre_query
\$pre_query = substr(\$pre_query, 0, -3);
\$keyWords = substr(\$keyWords, 0, -1);

// SQL statement to select the relevant documents
\$query = "SELECT DISTINCT d.title,i.index_term,i.weight,i.doc_id,d.full_text
FROM documents as d LEFT JOIN indexes as i ON i.doc_id=d.doc_id
WHERE \$pre_query ORDER BY d.title,i.weight";

// Execute the query and returns the results
\$result = mysql_query(\$query);

// If no result, error message is generated
if (!\$result)
echo mysql_error(\$link);

```
// Push the query result into an array
while ($row = @mysql_fetch_array($result)) {
    array_push($rows,$row);
}
```

```
$newWeight = 0;
$countSim = 1;
```

```
//process the $row array to choose the highest weight
for ($i=0; $i<sizeof($rows); $i++) {
```

```
    $element = current($rows);
    $curPath = $element['title'];
    $next = next($rows);
    $nextPath = $next['title'];
```

```
    if ($countSim==1) {
        $newWeight = $element['weight'];
    }
```

```
    if(strcmp($curPath, $nextPath)==0) {
```

```
        $bInSim = true;
        $countSim++;
        $newWeight += $next['weight'];
    }
```

```
    else {
```

```
        $bInSim = false;
```

```
        if($countSim > 1) {
```

```
            //if the documents appears $countSim time in the array,
            //multiply with $countSim to strengthen the weight of the
            //document that indicates its relevancy to the user query
            $finalWeight = $countSim * $newWeight;
```

```
        }
        else
```

```
            $finalWeight = $element['weight'];
```

```
            $countSim = 1;
```

```
    }
```

```
    if (!$bInSim && $finalWeight >= $threshold) {
```

```
        $element['weight'] = $finalWeight;
        $temp[] = $element;
    }
```

```
}
```

```
//Sort the weight from highest to lowest
foreach($temp as $line) {
```

```
    $weightSort[] = $line['weight'];
```

```
}
```

```
array_multisort($weightSort,SORT_DESC, $temp);
```

```
}
```

```
return $temp;
```