Using Keyword-Based Indexing for Desktop Content Navigation – Desktop Content Manager

By

Deepak Ravichandran

Bachelor of Technology, Naval Architecture (2001)
Indian Institute of Technology Madras

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the

Degree of Master of Science

Massachusetts Institute of Technology
June 2004

Author_____
Deepak Ravichandran
Department of Civil and Environmental Engineering
February 17, 2004

Certified by _____
John R. Williams
Associate Professor of Civil and Environmental Engineering
Thesis Supervisor

Accepted by _____
Heidi Nepf
Chairman, Department Committee on Graduate Students

Using Keyword-Based Indexing for Desktop Content Navigation – Desktop Content Manager

by

Deepak Ravichandran

Submitted to the Department of Civil and Environmental Engineering on February 17th, 2004 in partial fulfillment of the requirements for the degree of Master of Science

ABSTRACT

Researchers and students gather a wealth of information from the internet. With increasingly affordable large hard disk spaces, much content can be archived in the computer for future reference. However, the only means for organizing this information on today's machines is the folder system. Information on a topic may be spread over many folders. This provides for poor content navigation and retrieval.

This thesis describes how the concept of keyword-based indexing can be applied towards navigation of electronic content. The idea is implemented in the form of a software solution – Desktop Content Manager. The information on keywords and content are stored and represented in a manner facilitating automatic hyper linking of related information providing immediate access. The software was tested with sample content extracted from different sources and the scalability and limitations of the suggested approach identified. The findings are used to define further work in this area.

Thesis Supervisor: Prof. John R. Williams

Title: Associate Professor of Civil and Environmental Engineering

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor Prof. John R. Williams for his guidance and support throughout my stay at MIT. He has been a constant source of encouragement towards all my educational projects and research initiatives.

I would like to thank Abel Sanchez for the support extended by him in carrying out my research projects. I am also thankful to other members of IESL who have been a part of most of the research projects I have worked on. I am also appreciative of the timely administrative support provided by Joan McCusker throughout my graduate study at MIT.

At this point, I would like to express my thanks to Hari and Sivaram for their valuable suggestions. I am also thankful to my friends Anamika, Jeff and Byungkon Kim.

Finally, I would like to thank my parents, sister and Arthi for their love, support and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 – INTRODUCTION

1.1. MOTIVATION

A plethora of information is compiled and accessed as part of one's research study or course preparation. This information is spread over multiple, disparate information sources. Information archived on a topic appears in multiple contexts and entails a re-visit whenever related content is accessed. Information retrieval on a topic is not a trivial process due to the limited capacity of human cognition. Hence, timely access of appropriate information can aid in providing an enriched learning process.

The internet is being increasingly used as a medium to gather information on topics of interest. The information gathered from the web sites are electronically archived in files or the addresses of web pages bookmarked. Thus, the information representing a particular topic of interest to an individual is spread over a collection of accessed web sites and files present in the user's computer.

Techniques have been devised for tagging documents based on the content for easy retrieval in the future. However, the information that is being searched for might span over disparate information sources and the content retrieved in response to the information being searched might have references to other archived content.

Consider a typical classroom scenario where a student makes useful notes in his/her laptop during the lectures. This information is most often typed in a word processing software and saved for later reference. Whether the information is gathered from the web

or files in the file system or during a classroom lecture, they are most often related to each other and have some commonality between them. It is necessary that the content gathered from different sources be tagged appropriately and a single interface provided for their display and navigation.

Information accessed from one's computer can be in the form of files or web content or emails or any other information sources. In this thesis, files of selected formats and web content are considered. Content present in emails and other information sources that could be of potential use to the user are not considered. In the following section, some of the conventional practices for content management are discussed.

1.2.COMMONLY DEPLOYED PRACTICES

The vital components of a content management system are: Archival and Retrieval. A brief overview on the topics will follow.

1.2.1. Archival

"In history, since about 2500 B. C. archives can be found. *Archiving* is the process of collecting and managing objects with historical background. In general these documents are not prepared for publication (e.g. official documents) and are called dead documents, because these documents or objects never change. The term *archiving* describes also the classification of documents by formal methods (e.g. grouping, indexing, etc.). Indexing parameters can be subjects, locations and authors. Result of classification tends to an index summary in a printed and / or electronic version" [1]

In the context of research study or course work, even before computers were affordable for use as an electronic storage medium, content was typed or written down in papers and archived in folders. Typically, the key concepts involved in a research study or coursework were recorded. This approach made content management laborious and was gradually replaced by the electronic systems.

Electronic Archiving

"The term *electronic archiving* is similar to the concept of imaging (the process of digitizing, storage in databases and retrieval). In general, electronic archiving may be the first step to a digital office. Considering the electronic archiving process, it is subdivided into data storage, data retrieval, and data migration. The process of archiving depends on the characteristics of the document, which can be categorized in different ways: readability - human and / or machine-readable; type of storage media; document type (text, images, videos, music, etc.)"[1]

Some of the commonly deployed practices for electronic archival of information are:

1. Archiving files in File System

The information is stored in a set of files and these files are categorized by their parent directories, assigning directory names and file names based on the content and the context. However, with increasing number of files, the directory structure gets complicated and moreover, there is no means of representing links between the various documents.

2. Bookmarking websites

It is a common practice to bookmark the most important websites that were visited for future reference. However, only parts of the web pages may be relevant and in cases where the web pages are frequently updated, the information might be lost.

1.2.2. Retrieval

"Retrieval is the search for, and presentation of, archival material in response to a specific user request" (http://slim.emporia.edu/park/glossary.htm). In the context of computers, retrieval refers to the operation of accessing information from the computer's memory (http://www.thefreedictionary.com/retrieval).

Some of the common information retrieval practices are

Human Memory

"Retrieval of information is better if the information is organized in some manner supporting systematic search, such as in hierarchies"[2]. Based on the organization of files in the computer, the electronic documents representing a particular topic are accessed. These documents include information about bookmarked web pages. This assumes that the files have been systematically organized to facilitate easy retrieval.

Electronic search

With limits in human memory capacity, an electronic search is performed to retrieve archived documents representing the topic of interest. Lack of context representation of the documents retrieved makes it difficult to differentiate pertinent information from irrelevant information without browsing through the content. Expanding the domain of

search, the web is searched for information on the topic by means of a search engine. However, it is laborious to repeat the process and the content retrieved may not match with the information gathered on the previous attempt.

The practices for archival and retrieval are loosely coupled and focus on representing information only on individual topics of interest. However, the information stored in one's computer are most often related to each other requiring a system providing a navigational interface.

1.3. PROPOSED SOLUTION

The drawbacks of the commonly deployed archival and retrieval practices were discussed in the previous section. With increasing information on topics of interest, some of the practices need to be integrated to provide support for navigation. A content management software for the desktop - 'Desktop Content Manager' will address each of the issues discussed above, providing all the functionality needed. Some of the standard approaches are adopted with a significant amount of automation. The concept of keyword-based content archival has been applied to facilitate content navigation.

Irrespective of the format of the documents, the information present in them is most often in the form of text or figures. The content gathered from different sources enters the active database of the desktop content manager. The content is tagged either manually or automatically and links established between them. By this approach, whenever content on any particular topic is retrieved, the related topics of interest are presented displaying the

relationship level with each of the topics. There is no necessity to keep track of the location of documents or websites and the information is readily available when required.

The desktop content manager has been designed for the Windows operating system to demonstrate the suggested approach. This application has been designed to be an active component in the system tray of the operating system with convenient interfacing for upload, retrieval and navigation.

1.4. EXAMPLE SCENARIO

Consider an introductory course on programming as an example. Some of the introductory topics covered include types of variables, paradigms of object oriented programming etc. In the beginning of the course, a student comes across the concepts reference-type variable and value-type variable. The student looks at the lecture notes and browses the web pages until the information that is being sought is found. Either the student remembers all the information or stores them temporarily in a word processing software. In later part of the course, when more advanced topics are covered, there might be references to the concept of reference-type variable in some lectures. The student might now need to refresh the above concept. It would be very useful if the student is able to access the information on reference-type variable that he had collected in the past. On querying for information on the topic, the desktop content manager would provide all the information, the context in which they were stored and a listing of the sources for additional information.

1.5. RELATED INITIATIVES

WinFS: WinFS is the new storage system in Longhorn, with an improvement to the Microsoft Windows Platform in the following ways: Categorizing information in multiple ways and relating them with each other, providing common storage format for information collected on a daily basis and sharing of data across applications from different vendors [3]

In terms of storage, the functions provided by the desktop content manager would only be a subset of what WinFS would provide but the desktop content manager focuses more on navigation and the targeted audience is students and researchers. Apart from the support provided by the operating system for managing personal content, there have been several initiatives especially in the form of commercial software, targeting content management and information extraction from the web. However, most of the content management software provide tools for managing content at the server level targeting a group of users. There is also software in the market that does web extraction from one's favorite websites but most of them extract the web data and store them in the file system or a database without tagging them or linking them with other content in the desktop.

Offline Browsers: Offline browsers enable one to download an entire web site or a group of pages for offline viewing later. There are plenty of offline browsers available in the form of commercial software catering to a wide range of applications. Apart from the basic functionality of offline viewing of web content, some browsers provide more advanced features.

For students and researchers collecting information for their academic work, only parts of web pages from multiple web sites may be relevant. In such cases, instead of downloading an entire web page or web site, the user should be able to select the desired content among the set of web pages browsed.

Ideally, the features of content management, web extraction and content navigation should be available together. Content navigation is possible only if links can be established among the content gathered from different information sources. Desktop content manager is a light-weight application that addresses most of the above issues and is easy to use.

CHAPTER 2 – DESKTOP CONTENT MANAGER

2.1. KEY COMPONENTS

The key components of the desktop content manager include content extraction, keyword indexing, content retrieval and content navigation. Content extraction in this context refers to the process of extracting information from files in the computer, accessed web pages and classroom notes. Keyword indexing refers to the process of identifying the content with keywords that best describe the context. Content retrieval refers to the process of retrieving content from the active database by supplying keywords on topics of interest. Content navigation refers to the process of navigation among stored content.

2.2. CONTENT EXTRACTION

"Information extraction (IE) software is an important part of any knowledge management system. Working in conjunction with information retrieval and organization tools, machine-driven extraction is a powerful means of finding content on the Web. Information extraction software pulls information from texts in heterogeneous formats--such as PDF files, emails, and Web pages--and converts it to a single homogeneous form." [4]

Following a similar approach, the desktop content manager extracts content from selected formats and sources and converts them into plain text format for extracting more information. The prime sources for content extraction have been identified as -

1. Web Pages
2. Files – DOC, PDF, RTF and TXT

3. Class notes or Research notes

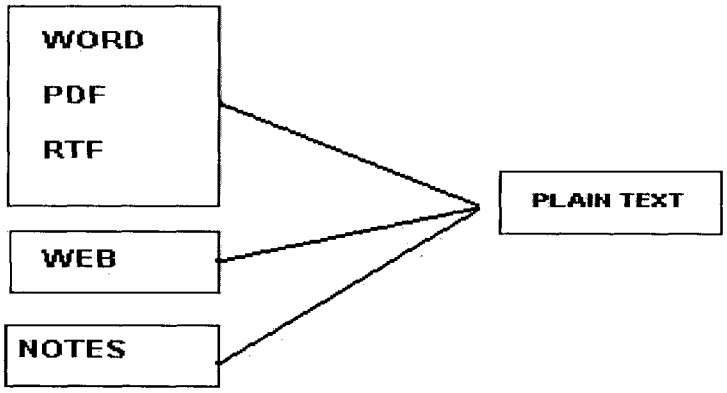


Figure 1 – Content conversion to plain text from selected file formats and sources

The extraction process is not automatic and the user selects the web content or the files that need to be indexed. Based on the content source, either the file path or the raw text extracted from web pages enters the database and is tagged based on the supplied keywords or identified keywords. Before any content enters the database, the text is automatically browsed to look for references to existing keywords in the database. If references to keywords are identified, this information is added to the database accordingly.

2.3. KEYWORD INDEXING

The content extracted needs to be identified by means of keywords to make future searches efficient. The keywords are either automatically extracted from the content by means of keyword extraction techniques or specified by the user. The keyword-content pair enters the database along with other related information. There are different modes

used for the keyword-content pairs based on how the keyword is related to the content. Keywords extracted from automatic keyword extraction need to be promoted to a regular keyword status as the keywords extracted by the algorithm needn't be exactly indicative of the context.

2.4. CONTENT RETRIEVAL

Information retrieval recovers a set of content from the database based on the entered keyword. This keyword has to match with the keyword that was used for tagging the content. For an individual, it is easy to identify the context in which the information was stored in the desktop content manager and hence retrieval is a simple process.

Currently, content and keyword are related at 3 levels - namely primary keyword, referenced keyword and probable keyword. Primary keywords are meant to be exactly indicative of the context of the content. Referenced keywords are keywords that are referenced in the content and probable keywords are automatically extracted keywords that may or may not represent the context.

On submission of a query, not only is the tagged content extracted from the database, but any content to which it draws a relationship with is also extracted. Relationship in this context means immediately referenced keywords as well as keywords appearing in content representing those keywords and keywords appearing in content referencing the queried keyword. The process is repeated till there are no more new keywords. Though the process does not follow a tree structure, the concept can be approximately represented

by means of a tree structure as shown in Fig. 2. Considering Fig. 2, starting with keyword K1, content corresponding to keyword K1 as primary and referenced keyword are accessed and the keywords referenced in them found to be K2, K8, K1 and K11. Since K1 has been already extracted, K1 is discarded and the process repeated with K2, K8 and K11. This approach is repeated skipping keywords already occurring in the tree until no more new keywords are found.

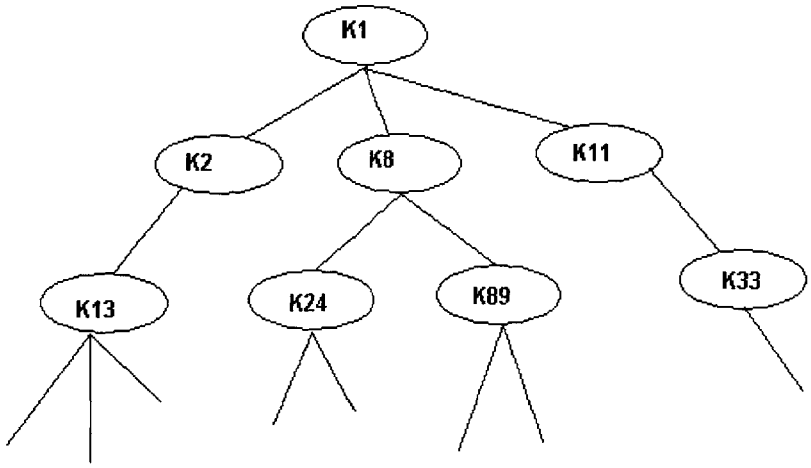


Figure 2 – An approximate representation of the extraction process for related keywords of K1

The mode information is extracted for all possible keyword-content pairs of the extracted keywords. They are represented in the form of a matrix with cells of colors corresponding to the mode of relationship. The matrix can be used to judge the context in which the content was stored and access any related content.

2.5. CONTENT NAVIGATION

Content archival and content retrieval systems should be suitably integrated to facilitate content navigation. Most of the time, when one is reviewing the information collected towards research work or preparing for an exam, a document is accessed more than once. Whenever there is a reference to some information presented in the document, the document is accessed again and read.

Content appears in multiple contexts entailing access from other related content. The hyperlinks in a web page as well as support provided by a web browser in page navigation make the process of web browsing effortless.

The desktop content manager supports navigation by:

- Providing access to content already read (Backward and Forward navigation)
- Hyperlinking web content and notes based on occurrences of keywords
- Providing links from the content-keyword matrix

Finally, the desktop content manager records the navigation of content to identify the context in which the content was accessed in the past.

CHAPTER 3 – SOFTWARE MODELING OF DESKTOP CONTENT MANAGER

3.1. SOFTWARE DESIGN

The software was developed using C# and .NET with XML and SQL Server 2000 providing database support. The software runs only on Windows Operating System as some of the functionalities provided by the Windows Operating System have been incorporated in the software. C# was chosen for coding purposes, identifying the functionalities provided by C# to interface with the Windows API. SQL Server 2000 was used for initial testing purposes and could be replaced by any other RDBMS or XML.

3.2. KEY SOFTWARE COMPONENTS

Before studying the details of software modeling and implementation of the desktop content manager, some of the software components that were vital for the functioning of the desktop content manager will be discussed

Some of the key software components are:

1. Regular Expressions
2. Windows Clipboard
3. Office Programming
4. XML as a storage medium
5. Embedding web browser in a windows form
6. Pdf-to-Text conversion

3.2.1. Regular Expressions

Regular expressions are used for matching patterns in strings. Instead of inputting the exact text that is to be matched within a string, the pattern of the text is described by means of regular expressions and the relevant parts extracted. A regular expression can be defined as pattern of text that contains standard characters and special characters known as Metacharacters. This pattern of text is to be matched when searching text [8]

Regular expressions provide flexibility and the process of searching dynamic text is made simpler. Some of the basic functions of regular expressions are [8]-

- Testing patterns in a string

An input string can be tested for the occurrence of patterns like a hyper link or an email address. This can also be termed as data validation

- Find and Replace text or pattern

After the patterns are identified in the input string, the occurrences can be removed or replaced with another text

- Extract a substring from a string based upon a pattern match.

One can find specific text within a document or input field.

For example, regular expressions can be used to search an entire website to remove outdated materials and replace some HTML tags. Each of the files are searched for occurrences of tags and the occurrences replaced [8]

In desktop content manager, regular expressions have been extensively used for

- Automatic keyword extraction
- Extracting information from html content
- Replacing keyword occurrences with hyperlinks
- Detecting the format of content

The System.Text.RegularExpressions namespace in C# provides properties and methods for creating and manipulating regular expressions.

3.2.2. Windows Clipboard

The windows clipboard APIs are used in the desktop content manager for transferring content from the web pages to the database. The clipboard is a set of functions that enable applications to transfer data. Because all applications have access to clipboard, data can be easily transferred between applications or within an application [9]

A memory object in the clipboard can be in any data format. Some of the data formats supported in Windows are: Rich Text Format (Rtf), Plain Text, Device Independent Bitmap (DIB), Html, Files etc. Applications define their own clipboard format by registering the format with the Windows clipboard. This is essential if the format of the word processing software is not supported by any of the standard data formats. The clipboard is user-driven and the clipboard transfers content from one application to another only in response to a command by the user [9]. The HTML clipboard format will be discussed in detail as any information extracted from the web is transferred by means of the clipboard.

HTML Clipboard Format

The following is an example of a clipboard:

The description of the clipboard provides information about the version number of clipboard and the start and end positions of the selected context and fragment.

Version	vv version number of the clipboard. Starting version is 0.9
SourceURL	The address of the website from which the content was transferred onto the clipboard
StartHTML	bytecount from the beginning of the clipboard to the start of the context, or -1 if no context
EndHTML	bytecount from the beginning of the clipboard to the end of the context, or -1 if no context
StartFragment	bytecount from the beginning of the clipboard to the start of the fragment
EndFragment	bytecount from the beginning of the clipboard to the end of the fragment
StartSelection	bytecount from the beginning of the clipboard to the start of the selection
EndSelection	bytecount from the beginning of the clipboard to the end of the selection

Table 1 – HTML clipboard description [10]

The URL of the web page along with the start and end positions of the selected text could be used in extracting selected parts from the accessed web pages at a later date.

Of the functions exposed by the Windows API for programming with the clipboard, the following functions are used in the desktop content manager:

RegisterClipboard	This function registers the application for accessing and storing information in the clipboard
GetClipboardData	This function returns the current content in the clipboard as a data object
SetClipboardData	This function copies content to the clipboard

Table 2 – Clipboard functions [9]

The C# libraries provide support for accessing these functions through the System.Clipboard namespace. The windows clipboard functions along with regular expressions are used in the desktop content manager for transferring relevant content from the web pages onto the database which will be explained in detail in the later sections.

3.2.3. Programming Office XP

The components of Microsoft Office XP – Word 2002 and Excel 2002 expose COM interfaces for programming purposes. Microsoft .NET and C# or Microsoft C++ with managed extensions can use these COM interfaces through the primary interop assemblies of Office XP that allow access of unmanaged COM code from managed .NET code. The Office XP primary interop assemblies need to be installed for programming against Word 2002 or Excel 2002 from Microsoft .NET. The desktop content manager

uses the libraries exposed by Office XP for opening and extracting text from a word document [11]

3.2.4. XML as a storage medium

XML (Extensible markup language) was used for storing formatted html content and any content exceeding the row size in a database. C# provides support for representing an xml file as a dataset and programming with the dataset.

3.2.5. Embedding Web Browser in a Windows Form

Microsoft web browser control is available as a COM component and can be added to the list of windows controls available in the Visual Studio toolbox. The web browser control was used for displaying the dynamically created html pages and other supported formats [15]

3.2.6. Pdf-to-Text Conversion

For conversion of Pdf files to text format, the pdftotext utility provided by Xpdf was used. Xpdf is an open source viewer for pdf files. The Xpdf project includes a pdf text extractor, pdf-to-postscript converter and other utilities [6]

The software components discussed above were used for initial testing purposes and can have alternative implementations.

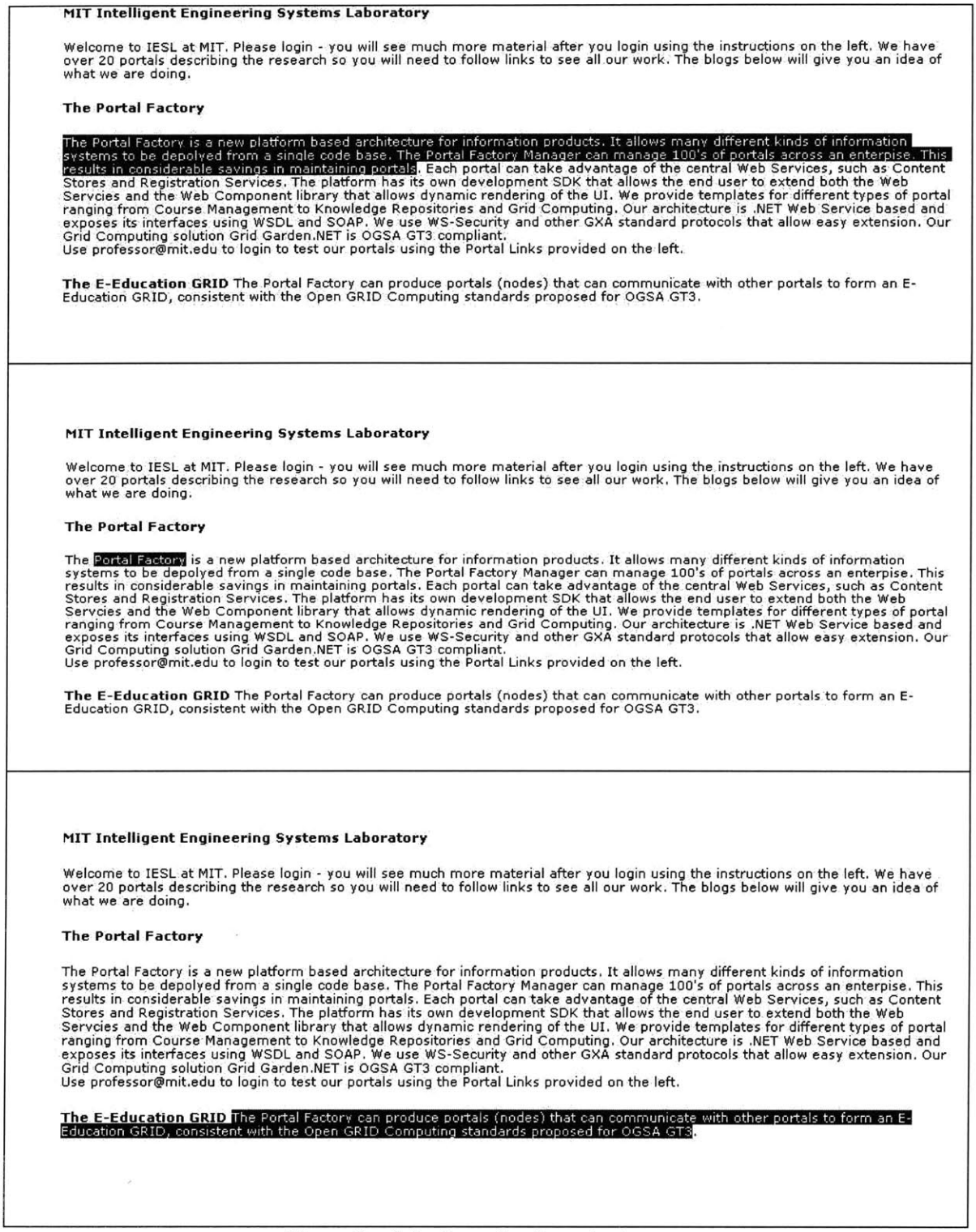
3.3. SOFTWARE MODELING

3.3.1. Content Extraction

The primary areas for content extraction are: The web, File system and Class notes or research notes

For context extraction from the web, the user selects the content from the web pages during browsing and copies it to the clipboard. In a similar fashion, more information is copied to the clipboard which may include content as well as keyword. In the case of file system, content is not extracted and stored again but the path of the file stored and the text version of the file browsed to identify keywords. The file information enters the desktop content manager either by a drag and drop operation or a file selection from the file dialog box opened from the desktop content manager. Currently, the following file formats are supported – PDF, RTF, DOC and TXT with some limitations. Finally, for recording notes, the notes are typed in the editor of the desktop content manager and the context or keyword information simultaneously entered.

The figure below (Fig.3) shows a sequence of selections made from a web page. First, a description on Portal Factory is highlighted and copied to the clipboard, then the keyword highlighted and copied to the clipboard and then more description on Portal Factory highlighted and copied to the clipboard. The set of content and the keywords can be copied in any order without specifying whether they represent the content or keyword as the Desktop Content Manager identifies the keywords based on the length.



MIT Intelligent Engineering Systems Laboratory

Welcome to IESL at MIT. Please login - you will see much more material after you login using the instructions on the left. We have over 20 portals describing the research so you will need to follow links to see all our work. The blogs below will give you an idea of what we are doing.

The Portal Factory

The Portal Factory is a new platform based architecture for information products. It allows many different kinds of information systems to be depolyed from a single code base. The Portal Factory Manager can manage 100's of portals across an enterpise. This results in considerable savings in maintaining portals. Each portal can take advantage of the central Web Services, such as Content Stores and Registration Services. The platform has its own development SDK that allows the end user to extend both the Web Servcies and the Web Component library that allows dynamic rendering of the UI. We provide templates for different types of portal ranging from Course Management to Knowledge Repositories and Grid Computing. Our architecture is .NET Web Service based and exposes its interfaces using WSDL and SOAP. We use WS-Security and other GXA standard protocols that allow easy extension. Our Grid Computing solution Grid Garden.NET is OGSA GT3 compliant.
Use professor@mit.edu to login to test our portals using the Portal Links provided on the left.

The E-Education GRID The Portal Factory can produce portals (nodes) that can communicate with other portals to form an E-Education GRID, consistent with the Open GRID Computing standards proposed for OGSA GT3.

Figure 3 – Selection of web content

(**Source**: http://ken.mit.edu/DevShell/DesktopDefault.aspx?tabindex=0&tabid=331)

The selected content from the web enters the active data list as soon as they are copied onto the clipboard. Content is appended to the data list until an upload option is clicked by the user. On clicking the upload option, the content from the data list is scanned and selected keywords are identified based on the length of the selected content. The format of the content is checked by means of regular expressions to differentiate keywords from content like hyper links or email addresses that have length of the same order. After identifying the format of the content and categorizing them as real content and keyword, real content enters the content list and the keywords enter the keyword list. In the case of files, the corresponding directory information and keyword information are appended to the content list and keyword list. For notes entered through the desktop content manager, the content and keyword information are typed in the editor and saved.

The figure below shows notes taken during a group meeting with the context elements being "Group Meeting" and "Group1"

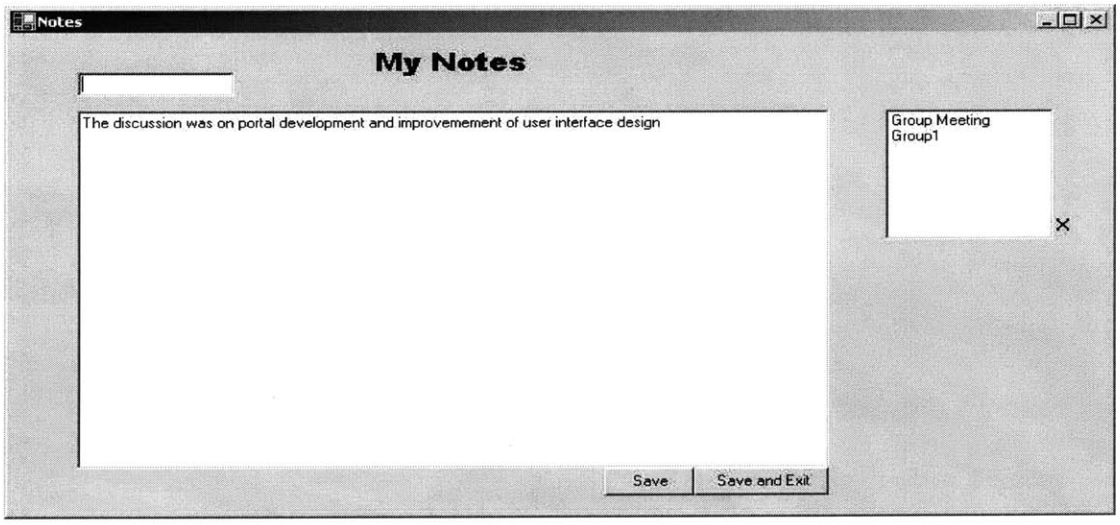


Figure 4 – Notes typed in desktop content manager

A content list is an array list with elements that are instances of the class content. The class content stores the following information about any content

Content	The text version of the main content that is extracted
Content Source	The source of the content which could be File system or notes or URL of a web page.
Content Type	The type of content – Text or File path or Email address or URL
Content Format	The format in which the original content was stored in the source which could Plain Text or Doc or Html or Rtf or Pdf
File	The file name where the archived web content is stored in html format
References	The keywords in the current database that the text in the content refer to
Length	Length of the text version of the content
Check-in Date	The date the content entered the database of the desktop content manager
Last Accessed	The date on which the content was last accessed (not needed during upload of content)
Auto Keywords	The keywords that were automatically extracted from the content

Table 3 – Content Representation

For information extracted from the web, by means of regular expressions, the source of the web page and the start and end of selected data is extracted. The text is browsed and searched for occurrences of archived keywords

3.3.2. Keyword Extraction

The keywords are either specified by the user or automatically extracted. For identification of the content by means of user-defined keywords, just like any other content, the keyword is highlighted in the web page and copied to the clipboard.

For automatic extraction of keywords, the algorithm proposed by Yutaka Matsuo and Mitsuru Ishizuka was used with customized modifications in parts of the algorithm. This algorithm for keyword extraction does not rely on a corpus and keywords are extracted from a single document. First, the frequent terms are extracted from the content after removing the stop words. Next a co-occurrence matrix is formed between these terms by counting frequencies of pair-wise co-occurrence. The idea is that a general term is used relatively impartially with every other frequent term whereas the frequent terms show co-occurrence especially with particular terms. [5]

Denoting the top ten frequent terms by G, "We denote the unconditional probability of a frequent term $g \in G$ as the expected probability p_g and the total number of co-occurrence of term w and frequent terms G as n_w. Frequency of co-occurrence of term w and term g is written as $freq(w, g)$. The statistical value of χ^2 is defined as

$$\chi^2(w) = \sum_{g \in G} \frac{(freq(w,g) - n_w p_g)^2}{n_w p_g}.$$

If $\chi^2(w) > \chi^2_\alpha$, the null hypothesis is rejected with significance level α. The term $n_w p_g$

represents the expected frequency of co-occurrence; and ($freq(w, g) - n_w p_g$) represents the difference between expected and observed frequencies. Therefore, large $\chi^2(w)$ indicates that co-occurrence of term w shows strong bias." [5]

In the context of desktop content manager, a similar approach is followed and the top 3 keywords in terms of the $\chi 2$ value are chosen and added to the auto keywords list of the content. The keywords obtained by this approach are reasonably indicative of the context and for the purpose of content management in a desktop, it is sufficient.

As the keyword extraction algorithm was only partially implemented for initial testing purposes, the software identifies only single words as part of the extraction process. For instance, for the selection process indicated in Fig.3, the keywords extracted were 'portal' and 'factory'.

3.3.3. Content Storage

The database for storing information about the content and keyword is relatively simple. Four tables are used in the database: one for the keyword, one for the content, one to represent the relationship between the keyword and the content and one to record the navigation history. (Data Model in Appendix – 2)

The content table stores the following information: Automatically assigned Content ID, Content in text format (if the size of the entire row doesn't the maximum limit), Type of

Content, Format of Content, Check-in date, Last access date and the File name where additional formats are stored.

The keyword table stores the following information: Automatically assigned Keyword ID, Keyword and Check-in date.

The keywordcontent table stores the following information: Keyword ID, Content ID, Check-in date, Last access date and the Mode. The mode is probably the only attribute demanding a brief explanation. The mode takes the following values: 1, 2 and 3. Mode 1 indicates that it is a primary keyword, a keyword which represents the content. Mode 2 indicates that it is a referenced keyword, a keyword which is referenced in the corresponding content and Mode 3 indicates that it is a probable primary keyword, a keyword which is usually generated by the automatic keyword extraction program.

The navigation table stores the following information: Automatically assigned Navigation ID, Path of navigation and the Keyword ID of the keyword which represents the navigation.

Thus, once the upload option is clicked, the following steps are executed in order:

1) The set of content in the content list enter the content table one-by-one and the content ID returned back

2) If the plain text format of the content exceeds the row limit, an xml file is generated and the text stored

3) The guid (Globally Unique Identifier) corresponding to the xml file is marked in the database

4) Additional formats like the html version of content is stored in the same xml file

5) The keywords in the keyword list enter the keyword table in the absence of duplication of entries and the keyword ID retrieved

6) The keyword ID and content ID are stored in the keywordcontent table with the mode being 1

7) The content is parsed through by the automatic keyword extraction program and the automatically extracted keywords once again undergo the same process like any other keywords. The keyword ID and content ID are saved with mode 3 if the keyword doesn't find any match with the existing entries

8) The content is parsed through to identify occurrences of keywords and if keywords are found, the content ID and the keyword ID are stored in the keywordcontent table with mode 2

9) Finally, the information stored in the buffer is freed and the program waits for another cycle

The process of extraction and archival discussed above doesn't use any novel techniques but was chosen to support the navigation pattern that will be discussed next.

3.3.4. Content Retrieval and Matrix Formation

The processes of retrieval and matrix formation have been clubbed together as they are closely tied to each other. The process of retrieval has the following steps

1) A keyword dictionary representing the keyword ID and keyword is loaded into the memory

2) On querying for information on a keyword, the set of content corresponding to the keyword as a primary keyword are loaded

3) Next, the set of content corresponding to the keyword as a probable primary keyword are loaded

4) Each content is parsed to look for occurrences of archived keywords with check in dates past the last access date of the content

5) A keyword-content matrix is generated taking into account every keyword and content stored in the database with the cells in the matrix taking the value of the mode of relationship. There are several empty cells indicating the fact that the corresponding keyword and content aren't related in any manner but for the convenience of navigation purposes and extraction of related content, this approach is adopted

6) The rows of the matrix represent the keyword ID's with a mapping from [1...M] to the corresponding keyword ID's. The columns of the matrix represent the content ID's with a mapping from [1...N] to the corresponding content ID's.

7) The matrix is crawled starting from the row corresponding to the keyword ID of the query keyword and as soon as a 1 or 2 is encountered, the matrix is crawled vertically. In a similar fashion, when on a vertical crawl, as soon as a 1 or 2 is encountered the matrix is crawled horizontally. The entire process happens recursively and the row and column indices are stored avoiding repetition of values. After the entire crawl is completed, a sub matrix is generated which has every keyword ID and content ID that is possibly related to the current keyword

8) After this program is executed, information about contents corresponding to the keyword ID's as primary keywords are pulled from the database and loaded

Thus, at the end of the retrieval process, information on every content that could possibly be of help in understanding the current content is loaded into the memory

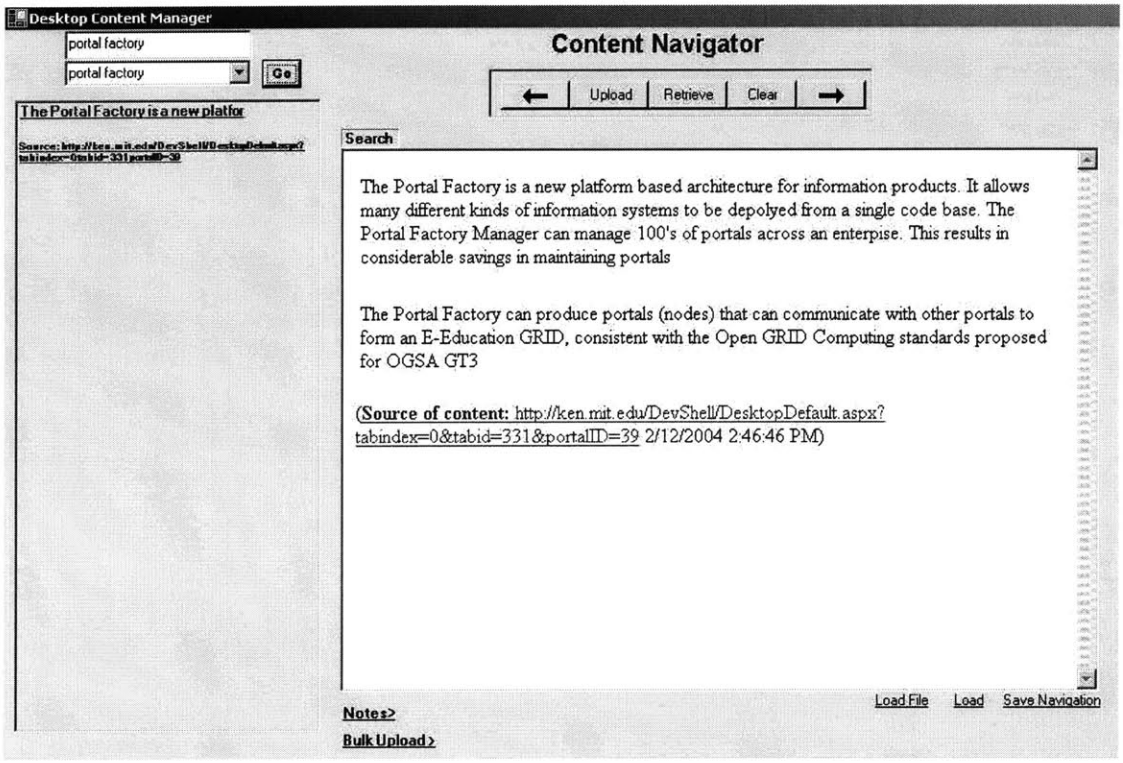


Figure 5 – Query for 'portal factory'

Fig.5 shows the content displayed in the content browser of the desktop content manager on querying for the term 'portal factory'. The content summary box in the left displays the summary in the form of the first few lines of the content followed by the source of the content. As more information about 'portal factory' is stored, they are automatically appended to the content browser and the summary information displayed in the content summary box.

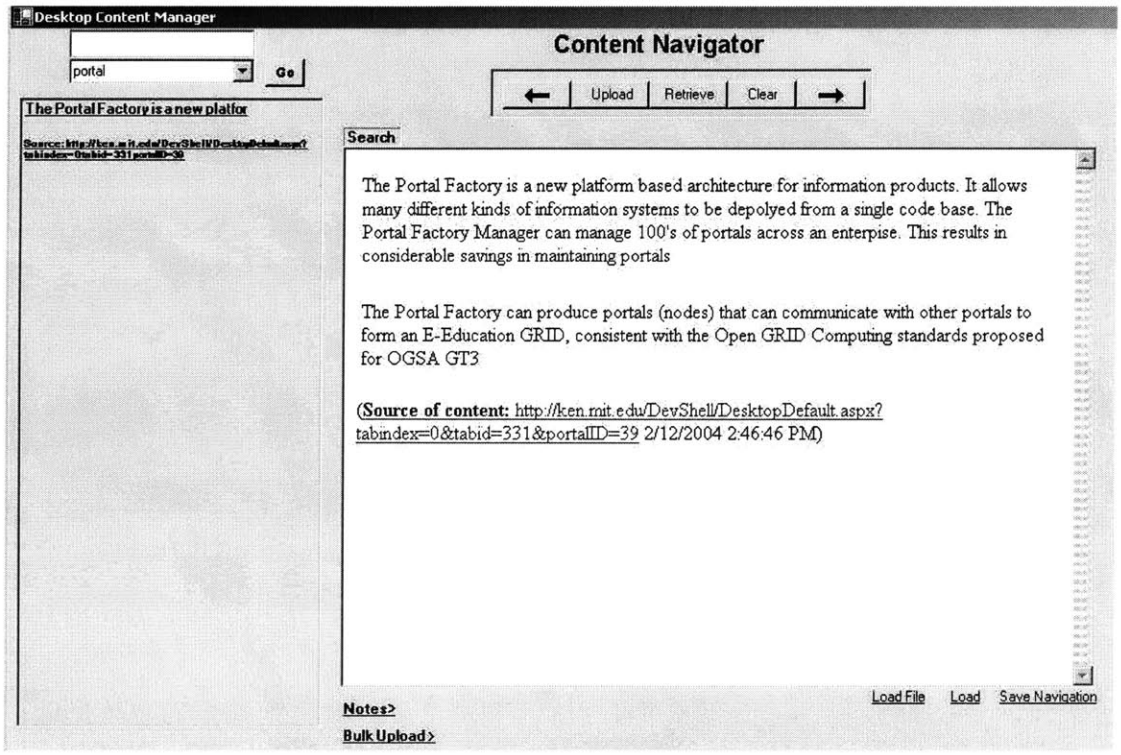


Figure 6 – Query for 'Portal'

On querying for 'portal', the summary list takes a different color indicating that it is a probable keyword for the content that was automatically extracted from the content and needs to be promoted. There are options to promote the keyword as a primary keyword and remove the keyword if there is no relation with the content

In figures 5 and 6, the keyword-content matrix is not shown which is otherwise displayed as part of a page in desktop content manager.

3.3.5. Content Display

The set of content corresponding to the keyword ID is displayed in a page. The page is composed of the following information:

Content List	Displays the set of content in the content browser which is a web browser control embedded in desktop content manager
Content Summary	Displays the first few lines of the content in the form of a link followed by a link displaying the source of the content which could be a URL or a file path. The summarized content links to the corresponding location in the content browser. The source link opens up the default browser with the web page loaded or the appropriate program with the file loaded. The importance of content for the keyword is distinguished from primary and probable primary by means of different color representations. Options are available to update probable primary to primary if the keyword indeed represents the content accurately
Matrix Box	This displays the sub matrix that was generated with the keywords representing the rows and the content representing the columns and the mode of importance (primary or referred) indicated by means of different colors

Table 4 – Anatomy of a Page

Thus, on querying for a keyword, the above information is displayed in a page

3.3.6. Content Navigation

The process of content navigation is as important as any of the above processes, if not more important. The desktop content manager relies heavily on the generated sub matrix for navigation purposes and navigation happens by means of pages. Before discussing the

navigation pattern, a brief discussion on the importance of generated sub matrix would be appropriate.

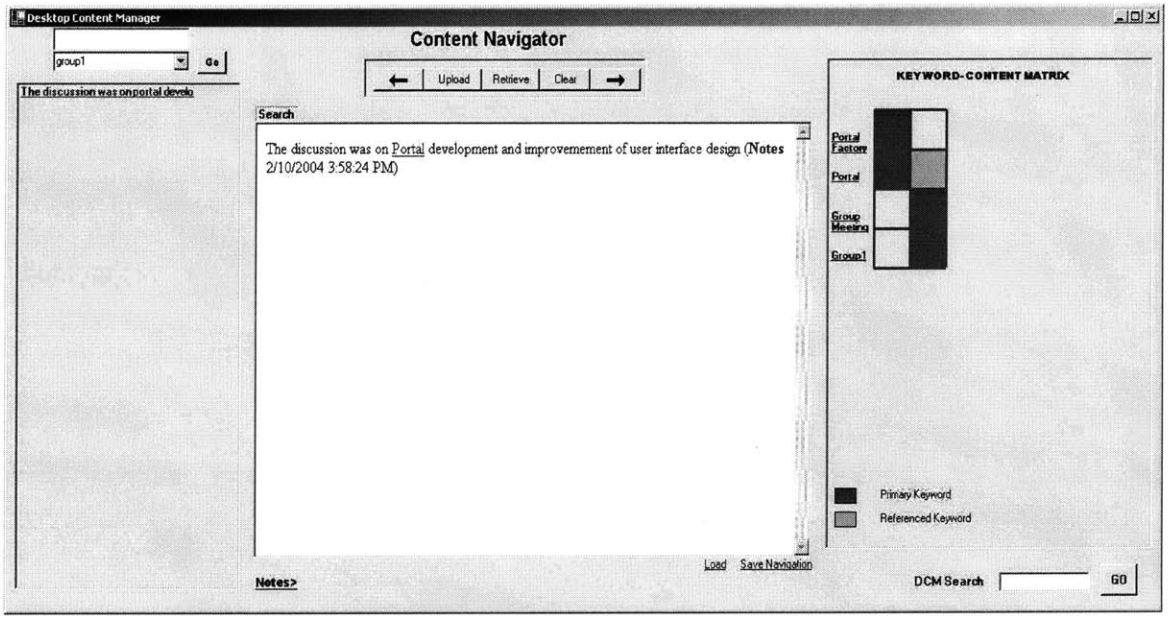


Figure 7 – Query for 'group1'

The above figure (Fig.7) shows how the display page looks on querying for the keyword 'Group1'. It is to be noted that this information was typed in the notes part of the desktop content manager. The keyword-content matrix displayed in the page is presented in more detail in Fig.8. The matrix has four rows and two columns with the cells being represented in different colors.

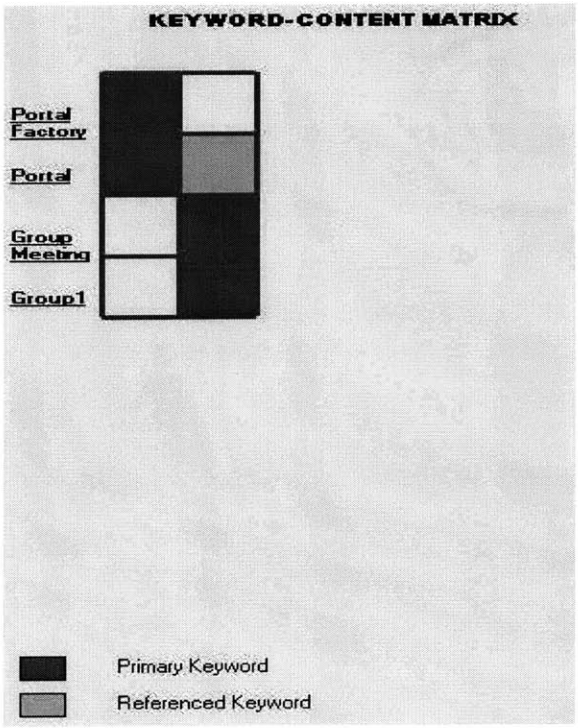


Figure 8 – Generated Keyword-Content Matrix

To highlight the importance of the sub matrix, the contents archived as part of the earlier examples are considered:-

On querying for the keyword 'group1', the set of content corresponding to 'group1' is displayed in the page with the relevant summary and the sub matrix. The sub matrix in Fig.8 shows how the keyword group1 is linked to other related keywords and content. The matrix displayed above has 2 contents and 4 keywords. The first content represented by the first column indicates that the content has two primary keywords 'Portal Factory' and 'Portal'. The second column indicates that the second content has two primary keywords 'Group1' and 'Group Meeting' and also references the keyword 'Portal'.

This could be very useful in the following cases:-

1) When going through the discussion that took place during the group meeting, more information on the term 'Portal' might be desirable

2) When a lot of content is displayed in the page and the user is looking for some content in particular, it should be easy to identify the content by looking at the relationship between the content and the keywords in the matrix

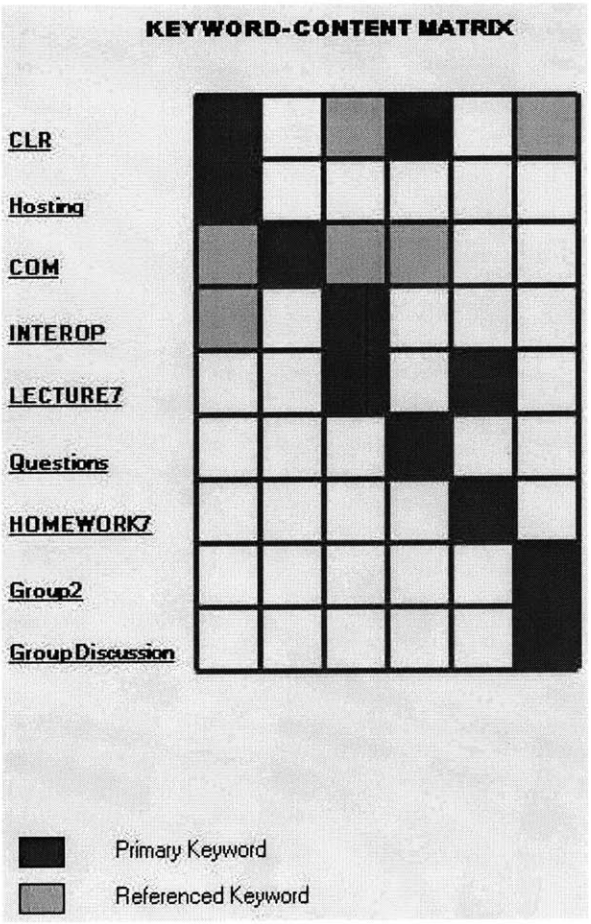


Figure 9 – Matrix-based navigation

Immediate navigation to content corresponding to a related keyword is achieved by means of a mouse-click on the keyword label that appears in the matrix. When there is a

lot of information presented on the keyword, to load the content the user needs, the user could just click on any cell on the appropriate column instead of looking at the summary information as the first few lines of the content do not really specify much about the content.

Considering Fig.9, the user queries for 'homework7'. Apart from the content presented on 'homework7', the entire matrix as shown in Fig.9 is displayed. This can be very useful to trace all the information that might be useful towards doing that homework. For instance, considering the last column in the matrix, the user identifies that homework7 has some relation with lecture7. Next, looking at the content corresponding to lecture7, he is presented with information on the terms 'COM' and 'CLR' with hyperlinks generated on them. These hyperlinks can be used only for navigation to immediately referenced content but by means of the matrix, the user can also figure out that there was a discussion on the topic of 'CLR' during one of the group discussions he had with one of his groups.

Thus, the page is in turn a class with the following information:

Content List – An array list of content

Content Summary List – An array list of link labels of summary and source of content with information about their location in the current page

Keyword – Keyword representing the current page

Forward and backward navigation is achieved by means of storing the pages in a page list which is an array list of pages. On querying for a new keyword, more pages are added to the page list. Although, the sub matrix allows for navigation only among related content, the forward and backward navigation can extend to previous queries and later queries

3.3.7. Generating HTML

It has been discussed earlier that on querying for a keyword, the set of content corresponding to the keyword is displayed in the content browser. For information gathered from the web or through the notes part of the desktop content manager, an html page is dynamically generated by reading the formatted content data from the xml files corresponding to the set of content in the content list. Hyperlinks are automatically generated whenever there are references to other keywords. The html page is destroyed when it is no longer in use. In the case of files, they are displayed in their native formats in the content browser. Alternatively, the text version could be displayed and hyper links generated. Fig.10 shows how the content displayed on querying for 'Group1' or 'Group Meeting' hyperlinks to the information stored on 'Portal'

Thus, the desktop content manager supports content navigation by:-

1) Matrix-based Navigation
2) Direct link from the content summary
3) Generating hyper links for web/notes content
4) Forward and Backward Navigation in terms of pages

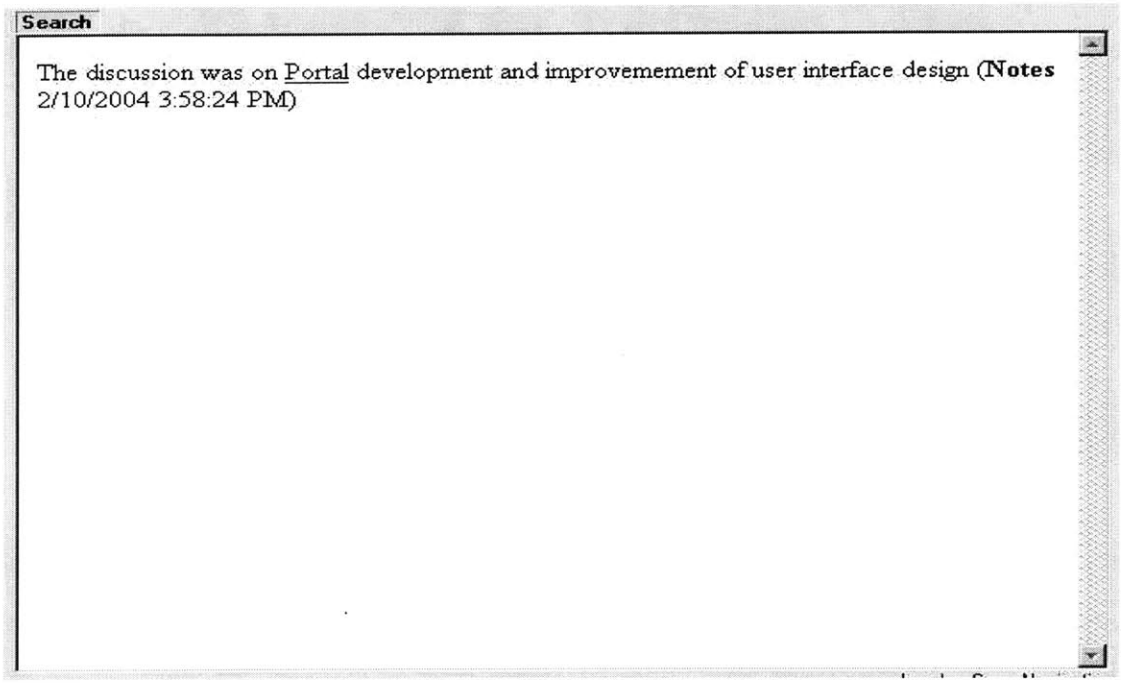


Figure 10 – Automatic generation of hyper links in the content

The source of content is provided in the content browser at the end of the content along with information about the date and time of check-in.

3.3.8. Recording Navigation

The pattern of navigation on a subject can be recorded at the end of navigation. For instance, preparing for a quiz on programming, if one browses the content topic by topic, navigation on each of the topics can be recorded and the cumulative navigation can be recorded as a set of navigations. Examples of navigation patterns recorded in the database are as follows:

123K23K45C – First access all content corresponding to the keyword with keyword ID '123', then access all content corresponding to the keyword with keyword ID '23' and finally access content with content ID '45'

13C7N – First access content with content ID '13' and then access entire navigation corresponding to navigation ID '7'

On loading the navigation at a later date, the navigation pattern can be browsed from start to end in the content browser. Simultaneously, a keyword search can be performed and the pages added to the existing navigation at any point of the navigation. Based on the changes made to the current navigation, changes are made to any other navigation which is a part of the current navigation. For instance, considering the pattern '13C7N', if a page is added to the navigation somewhere in the middle, the navigation pattern with ID '7' is also updated.

Thus, the processes described in the chapter were adopted from the stage of extraction to navigation. Some of the processes discussed above might need further revision and addition of more elements. The software modeling will be studied again in the last chapter defining limitations in the current approach and scope for further work in this domain.

CHAPTER 4 – EXTENDED APPLICATIONS OF DESKTOP CONTENT MANAGER

4.1. PRIMARY OBJECTIVE

The primary objective of the desktop content manager is to facilitate the learning process for students and researchers gathering content from different information sources. The desktop content manager helps them retrieve content at ease and provides them with a single stop-place for accessing archived information on any topic. Support for navigation within the archived content and saving the navigation for future reference is provided.

The applications of desktop content manager do not confine to the above areas alone but stretch beyond that to the following areas –

1. Matrix representation for course lectures
2. Context based searching
3. Restricted web site search

4.2. MATRIX REPRESENTATION OF COURSE LECTURES

The concept of content-keyword matrix could be extended to representing a set of course lectures in the form of a matrix depicting their association with the keywords. To highlight the same, a separate piece of code was written to download course lectures in pdf format from a course website and upload the information into the desktop content manager. A common context was given to lectures downloaded from a particular website and the automatically extracted keywords were promoted to primary keyword status. The content was not searched for references.

The code was tested on MIT OpenCourseWare website for 1-221J (Transportation Systems, Fall 2002) and the first 6 lectures considered [16]. The following information was entered in the software:

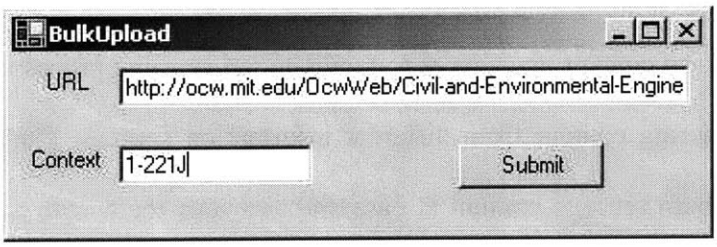


Figure 11 – Upload of course lectures for matrix generation

The desktop content manager was queried for "1-221J" after the upload process. The following matrix was generated after removing an unrelated keyword.

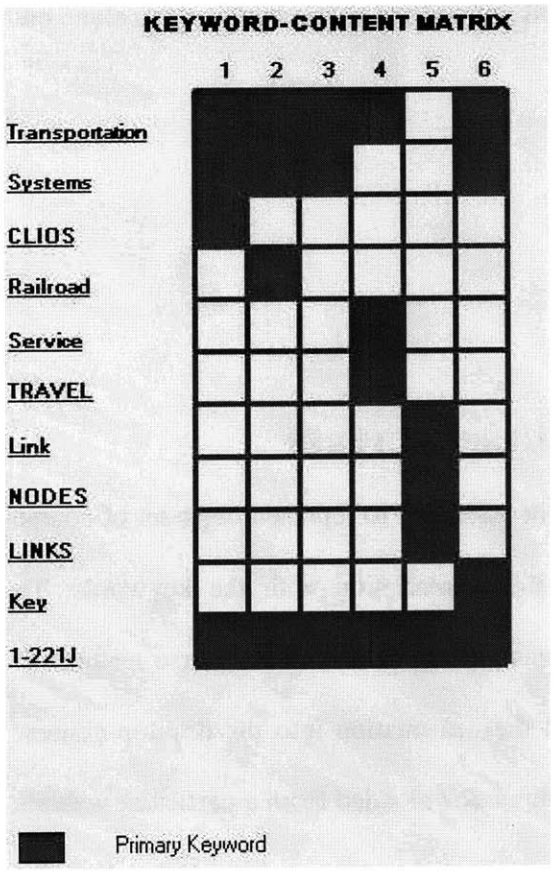


Figure 12 – Generated Matrix for the first 6 course lectures

As shown in Fig.12, the columns numbered 1-6 represent lectures 1-6 in that order. The rows are represented by the automatically extracted keywords along with the inputted context. Though the extracted keywords may not be the prime keywords representing the lectures, most of the extracted keywords are reasonably indicative of the content in the documents.

4.3. CONTEXT BASED SEARCH

As discussed in the first chapter, the web is one of the largest repositories of information and a popular stop place for information extraction. To extract information on any particular topic, a keyword based search in any of the search engines is a standard practice. In some cases, the results of query based searching match exactly with the information one is looking for. However, there are cases where a lot of information is presented and the information one is looking for does not appear anywhere in the first few search results. The reason is not due to unavailability of information but is most often due to lack of context representation in the query. For instance, say in the context of software, a search is made on component object model with the query string being COM. However, the first few search results are not even close to the topic of component object mode. This is because the term COM appears in more than one context and the query string needs to be modified accordingly based on the context of the topic.

The information stored in the database of the desktop content manager could be utilized towards re-designing the query. For instance, considering the matrix generated above, it is easy to identify some level of hierarchy. It can be figured out that 'links', 'nodes' and 'travel' are sub-topics in transportation systems. When querying for 'links' or 'nodes',

the context information of 'transportation systems' might be useful. On careful investigation, more inferences can be made about the relationship between the keywords. Though, a true hierarchy of information is not going to exist, at the least, some level of hierarchy can be identified from the returned matrices.

Thus, in the re-design of the query, which includes the context in which the keyword appears, the information stored in the database of the desktop content manager could be of use.

4.4. RESTRICTED WEB SEARCH

Besides using the search engines, a localized search on the websites of interest is a standard practice. For instance, after having identified the list of web sites that is most suited to one's research or course work, a localized search for information on each of the web sites is common.

The desktop content manager stores the URL of the web pages along with the user selected information. Assuming that a simple search engine could be written or any of the available search engines be given the list of web sites to be searched for, the list of accessed web sites for information on a particular topic is available in the database of desktop content manager. Thus, when looking for information on a keyword, the web sites corresponding to all related content could be first searched for before performing a search on the entire web repository.

For the sake of testing purposes, the web APIs offered by google were used for searching specific sites on particular topics. Google provides access to the web APIs for interested developers with the restriction of 1000 queries per day [12]. A site specific query looks like: <Query terms> Site: <URL> [17]

4.5. OTHER APPLICATIONS

Other than the applications discussed in this chapter, the desktop content manager could be used for more applications in the academic field. The desktop content manager could be used for preparing term papers where a lot of literature survey is done to collect information from different sources. When studying for exams, the desktop content manager could be used to save navigation of accessed content in the order of access. Content management being an active area of research in educational institutions as well as business organizations, the features of desktop content manager could be extended to support new applications.

CHAPTER 5 – CONCLUSIONS

5.1. SUMMARY

The desktop content manager has been designed to archive and index content extracted from the web, file system and recorded notes. The contents were indexed by means of supplied keywords as well as automatically extracted keywords. The contents were linked with each other based on the indexed keywords and referenced keywords. On querying for a keyword, the set of content directly representing the keywords as well as information about all supporting content were extracted. The keyword-content relationship was represented in the form of a matrix to allow for easy identification of the context in which the contents were stored. The contents were displayed in a page along with their summary and source information. The contents extracted from the web and notes taken through the desktop content manager were automatically hyper linked to provide immediate access to any related content. The matrix was enabled with links to any related content. Facility for saving the navigations was made available.

Thus, there was no necessity to keep track of the path of files or the URLs of the accessed web pages containing useful information. Desktop content manager provided a single interface for accessing all archived content representing a particular topic. Navigation between content was made possible for web content and notes recorded through the desktop content manager. Multiple file formats supported by the software were displayed in the content browser. Finally, the navigation patterns on topics could be recorded for future access.

The limitations identified in providing the features discussed in the previous section will be discussed in the following section.

5.2. LIMITATIONS

The desktop content manager required an active participation from the user. It scaled best for content extracted from the web as the additional processing involved was relatively simpler compared to that involved for the files. Navigation by means of automatic hyper linking was possible only for web content and notes recorded through the desktop content manager. The supported file formats required the respective components to be pre-installed in the machine.

As the size of the extracted data increased, the processing became slower but new threads were spawned allowing for continuation of the main process. Wait in the order of a few seconds was necessary for viewing content that was just uploaded.

The matrix displayed was most useful only when there were enough links between the contents. However, with increasing number of links, the dimensions of the matrix became too large. This issue can be approached by designing a scheme for specifying the extent to which links can be generated. Relative importance of keywords representing a particular content was not displayed.

The objective of the desktop content manager was to visualize the working of a keyword-based indexing system for navigation purposes and investigate the applicability of the

system. Many new features in the form of support for media and diagrams, weighing system for keywords, clustering of content could be provided and the processing time improved by implementing the processes more efficiently. Further work possible in this domain will be discussed in the next section.

5.3. FURTHER WORK

5.3.1. Weight-based keyword-indexing

The relative importance of keywords representing a particular content could be judged by means of assigning weights for the keywords. The statistical parameters specified by the automatic keyword extraction process as well as dynamic parameters like number of accesses could be incorporated to provide a realistic weight-based system.

5.3.2. Automatic Classification

The extracted keyword-content information could be used to classify the content by means of topics, number of accesses, dates of check-in and alerts could be issued for content not accessed in a while prompting the user for action on the content.

5.3.3. Additional file formats

Support for more file formats could be provided and the notes section of the desktop content manager improved allowing the user to sketch diagrams and edit content. Media files could be indexed by extracting the audio information by means of automatic speech recognition. The selected file formats could be studied further to enable direct navigation from the files.

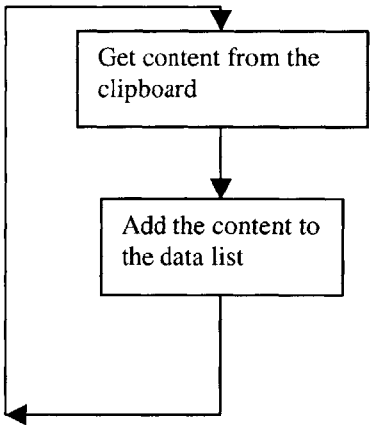
Finally, more options could be provided as part of the user interfaces to manage content and processes involved in the implementation of the software made more efficient in terms of memory usage and processing time.

APPENDIX 1 – PROCESS FLOW CHARTS

Some of the primary processes in the desktop content manager will be covered in this section. The important steps are highlighted without getting into the details.

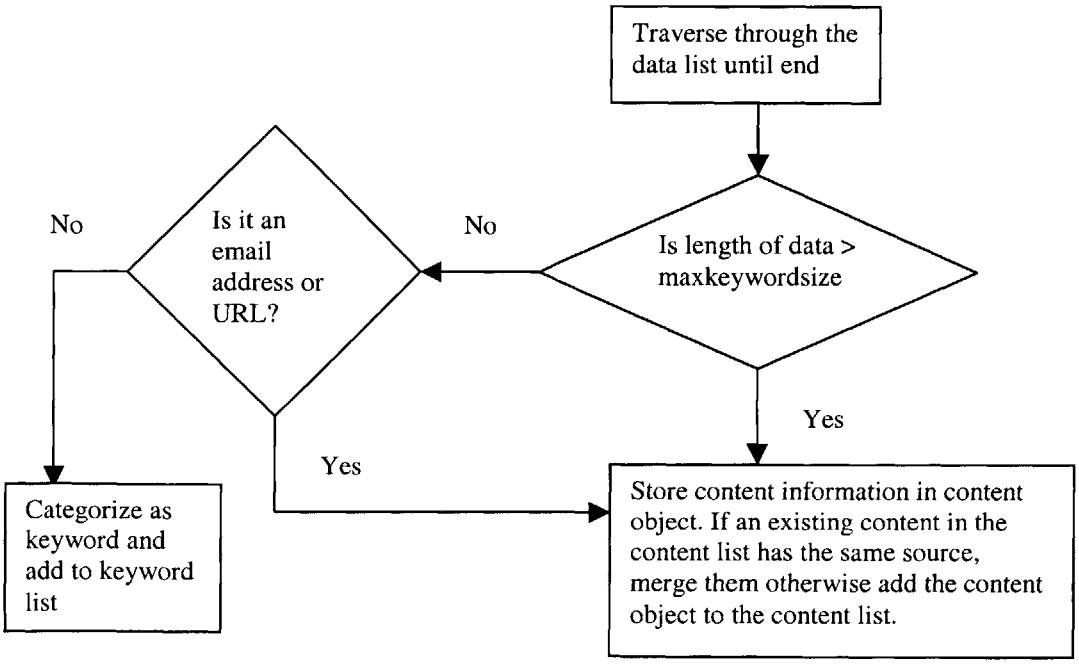
PROCESS BEFORE UPLOAD (FOR WEB CONTENT)

When in active mode, the process copies any content copied to the clipboard and adds it to the data list.



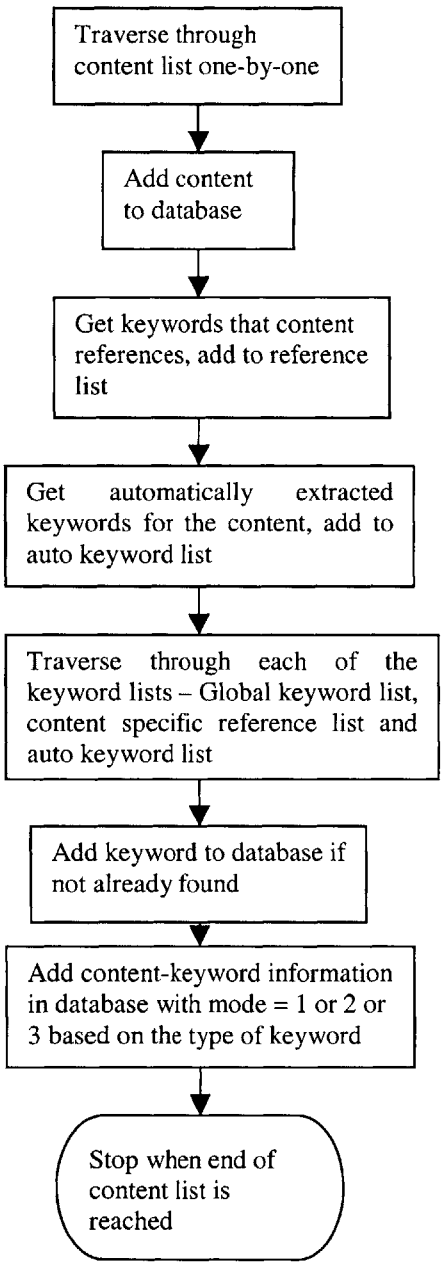
On upload, the data is first categorized in the case of web content

CATEGORIZING BEFORE UPLOAD (FOR WEB CONTENT)



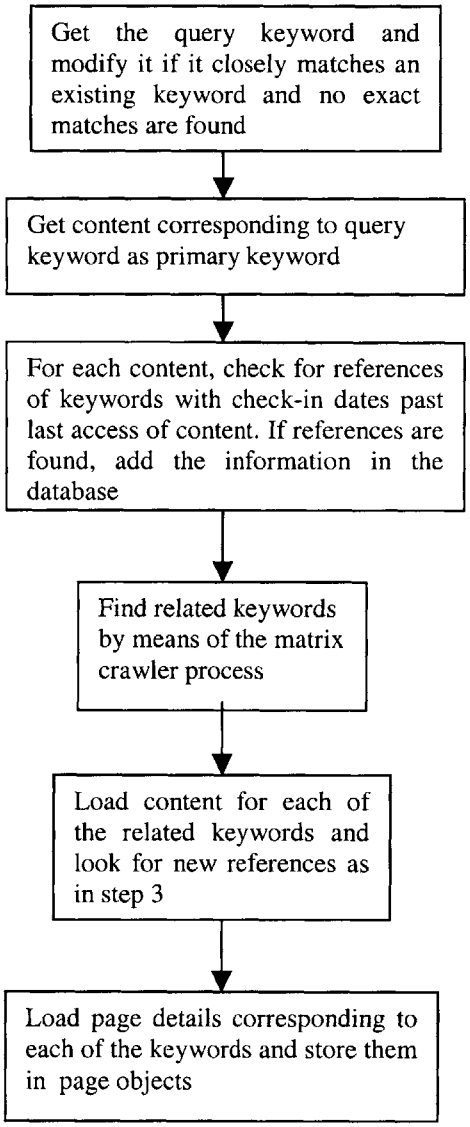
UPLOAD PROCESS

As discussed in chapter 3, once the upload option is clicked, the following steps are executed. The mode of relationship of a keyword-content pair entering the database depends on how the keyword is related to that particular content

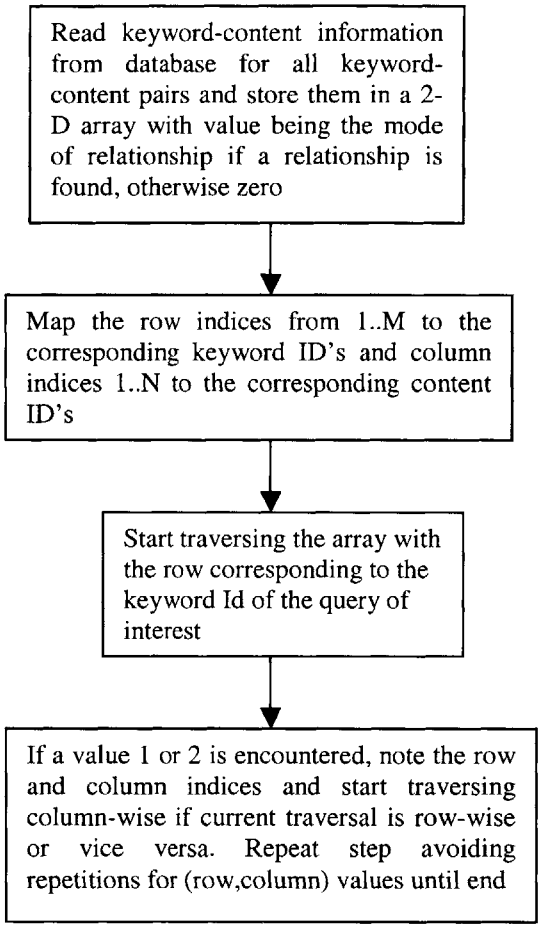


RETRIEVAL PROCESS

Retrieval process involves a lot of steps and is followed by page generation and page display. Only some of the important steps are highlighted here.



SUB-MATRIX GENERATION

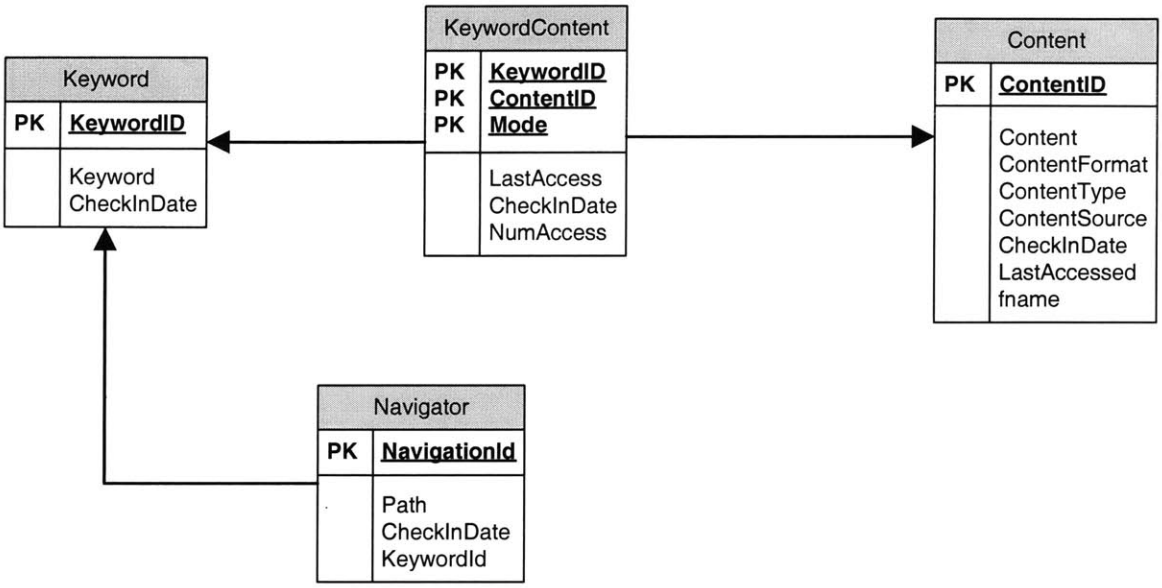


The function for deriving the sub-matrix follows:

```
cellList - ArrayList of objects of type cell which stores row and
column information of all keyword-content pairs related to the query
keyword

Matrixe - The 2-D array with mode values for all possible keyword-
content pairs

private void deriveMatrix(int dim, int key, int numrow, int numcol)
{

int startrow = 0;
int startcol = 0;

// dim refers to dimension to indicate whether to perform row-wise
traversal or column-wise traversal

if ( dim == 1 ) // Row-wise traversal
{
      if ( key != -1) startrow = key;
```

```
            else
            {
            // Get the row position from previous traversal
            Cell cell = (Cell) cellList[cellList.Count - 1];
            startrow = cell.row;
            }

for (int i = 0; i < numcol; i++)
 {
if ((matrixe[startrow,i] == 1) || (matrixe[startrow,i] == 2))
   {
      Cell cell_ = new Cell();
      cell_.row = startrow;
      cell_.col = i;
      cell_.weight = matrixe[startrow,i];

      // Check for existence of cell values
      if (absent(cellList,cell_))
      {
      // Add the row and column information to the cell list
      cellList.Add(cell_);
      // Start traversing column-wise from the current row position
      deriveMatrix(0, -1, numrow, numcol);
      }
   }
 }
}

else

// Column-wise traversal

{
// Get the column position from previous traversal
Cell cell = (Cel) cellList[cellList.Count - 1];
startcol = cell.col;

for (int i = 0; i < numrow; i++)
  {
if ((matrixe[i,startcol] == 1) || (matrixe[i,startcol] == 2))
    {
      Cell cell_ = new Cell();
      cell_.row = i;
      cell_.col = startcol;
      cell_.weight = matrixe[i,startcol];
      if (absent(cellList,cell_))
      {
      cellList.Add(cell_);
      deriveMatrix(1, -1, numrow, numcol);

      }
    }
  }
}
}
```

APPENDIX 2 – DATA MODEL



The database consists of four tables and six stored procedures. A description of each of the tables is covered in the section on software modeling in chapter 3.

REFERENCES

[1] Christian Gütl and Wilfried Lackner, "*WebSave - Archiving the Web for Scholar Work*", SITE 2001 (http://www.iicm.edu/iicm_papers/websave/websave.pdf)

[2] *Memory Encoding, Storage Retention and Retrieval* (http://brain.web-us.com/memory/memory_encoding.htm)

[3] *Books: Chapter 4: Storage (Introducing Longhorn for Developers)* (http://msdn.microsoft.com/longhorn/understanding/books/default.aspx?pull=/library/en-us/dnintlong/html/longhornch04.asp)

[4] Katherine C. Adams, *ONLINE, March 2001 | The Web as Database: New Extraction Technologies and Content Management* (http://www.onlinemag.net/OL2001/adams3_01.html)

[5] Yutaka Matsuo and Mitsuru Ishizuka, "*Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information*", FLAIRS'03, 2003 (http://www.miv.t.u-tokyo.ac.jp/~matsuo/papers/flairs03.pdf)

[6] *Xpdf: About* (http://www.foolabs.com/xpdf/about.html)

[7] *Regular Expression HOW TO* (http://www.amk.ca/python/howto/regex/regex.html#SECTION000310000000000000000)

[8] *Uses for Regular Expressions* (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/jscript7/html/jsreconregularexpressions.asp)

[9] *About the Clipboard* (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/dataexchange/clipboard/abouttheclipboard.asp)

[10] *HTML Clipboard Format* (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/dataexchange/clipboard/htmlclipboardformat.asp)

[11] *Programming Microsoft Word 2002 and Excel 2002 with Microsoft Visual C#* (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnexcl2k2/html/odc_offcs.asp)

[12] *Google Web APIs – Home* (http://www.google.com/apis/)

[13] *Create a Windows Clipboard Monitor in C# using SetClipboardViewer* (http://www.radsoftware.com.au/web/CodeZone/Articles/ClipboardMonitor.aspx)

[14] *MSDN Library* (http://msdn.microsoft.com/library/)

[15] Ted Faison, *Component Based Development with Visual C#,* M&T Books, 2002

[16] *MIT OpenCourseWare | Civil and Environmental Engineering | 1.221J Transportation Systems, Fall 2002 | Lecture Notes* (http://ocw.mit.edu/OcwWeb/Civil-and-Environmental-Engineering/1-221JTransportation-SystemsFall2002/LectureNotes/index.htm)

[17] *Google Help* (http://www.google.com/help/refinesearch.html#domain)