

LOW BANDWIDTH INFORMATION RETRIEVAL SERVICES FOR RURAL AREAS

Group Name: SHAs

Members: Manjusha Nair & Jisha R C

Department of Computer Science and Applications,

Amrita University, Amritapuri Campus. Amritapuri, Kollam, Kerala, India.

ABSTRACT

The project titled “INFORMATION RETRIEVAL SERVICES” is a system intended to make search over internet more efficient, fast and focused. This is an information retrieval system which provides the user with relevant and authentic results. This is a system designed to make search more efficient in low band width networks.

The Project acts as an efficient and authentic content retrieval system by performing three distinct operations namely Local search, Focused crawling and Content adaptation. This performs local search to provide the data to the user from the storage server which contains the already downloaded pages which optimizes the search and also helps to utilize the low bandwidth networks efficiently. Focused crawler provides the authentic and relevant data according to the context. The updation of storage server is performed by the crawler. The system downloads the contents of relevant urls and stores the content adapted version of data in its storage server. The system will increase the speed of searching by reducing complexities, increasing the efficiency and providing more accurate results.

The Local Search and Content Adaptation part of the system is developed using the C#.Net & Focused Crawler is developed in Java. The communication between the modules is achieved through the messaging services ActiveMQ in Java and ActiveNMS in C#. The Local search

module uses the Lucene library. The Wordnet Dictionary and the google suggest is used to find to get the synonyms of the keyword.

OBJECTIVE AND SCOPE

The Objectives of the system are:

- Internet Search Optimization
- Fast & Efficient Content Retrieval
- To provide authentic and relevant data with economical bandwidth usage
- Search can be managed according to the user query and preferences
- To provide a more effective web search over low bandwidth networks.
- To provide a more user friendly design to an internet searching.
- To provide quickness for the retrieval of documents.
- To reduce the greater search time for the user.

The objective is to develop a new system which will increase the speed of existing system by reducing complexities. It is a system which is intended to support effective web search over low bandwidth networks.

Development Tools used

Net Beans 6.1

Java 5.1

Visual Studio 2005

C# .NET

ActiveMQ

Apache NMS

Microsoft Office Access 2007

SYSTEM STUDY

Current system

When a user enters a query into a search engine (typically by using key words), the engine examines its index and provides a large listing of best-matching web pages according to its criteria, usually with a short summary containing the document's title and sometimes parts of the text. The existing system of search engines employs the general purpose Web crawler which gathers as many pages as it can from a particular set of URL's available. The rapid growth of the world-wide web poses unprecedented scaling challenges for general-purpose crawlers and search engines. The existing system collects and indexes all accessible web documents to be able to answer all possible ad-hoc queries. Furthermore, the imminent explosion of web will challenge even the most scalable solutions. Even though the focused crawlers are available now a day their working in low bandwidth networks is not efficient.

The usefulness of a search engine depends on the relevance of the result set it gives back. While there may be millions of web pages that include a particular word or phrase, some pages may be less relevant, popular, or authoritative than others. How a search engine decides which pages are the best matches, and what order the results should be shown in, varies widely from one engine to another. The methods also change over time as Internet usage changes and new techniques evolve.

Disadvantages of Existing system

- Automated method of collecting information is rather crude
- Information can be out of context
- May produce out of date sites
- Often funded by advertising
- No effective web search over low bandwidth networks.

Proposed System

The proposed system has many facilities and options like search optimization, expanded search query interface with preferences to best satisfy the network bandwidth. Our search engine is an information retrieval system. This enables users to perform web search asynchronously and find what they are looking for in one round of intermittency as opposed to multiple rounds of search/downloads. Users get the aggregated details only from authentic sites. Since an efficient local caching is done the total internet bandwidth usage can be reduced highly. The proposed system includes the modules like focused crawler, local search and content adaptation.

1. Local search

The local search is a searching process within the storage server which contains the downloaded pages.

When the user enters the search query in the text field available, the system performs a lookup in the keyword_store table to find whether the given keyword is present in the storage server. If the query string is present in the storage server, the keyword is searched in the index and the relevant contents are retrieved and presented to the user. If the query string is not in the storage server, then the system sends the search query for crawling. I.e. The system searches the query string in the web and retrieves the pages relevant to the given query. Then the content of the page is adapted and stored in the storage server. The next time, when the user searches for the same query string, the contents are retrieved from the storage server by the process of local search. This local search mechanism is time saving and efficient.

The user can also specify direct URLs to in the search textbox which will fetch the page directly.

2. Focused Crawler

A general purpose Web crawler gathers as many pages as it can from a particular set of URL's. Where as a focused crawler is designed to only gather a document on a specific topic, thus reducing the amount of network traffic and downloads. The goal of the focused crawler is to selectively seek out pages that are relevant to a pre-defined set of topics. Rather than collecting and indexing all accessible web documents to be able to answer all possible ad-hoc queries, a focused crawler analyzes its crawl boundary to find the links that are likely to be most relevant for the crawl, and avoids irrelevant regions of the web.

There are three functions for the crawler . The strategy used is Breadth First Search(BFS). First, automatic crawler, which will continue crawling in the web and fetch URLs relevant to the pre-organized contexts. Second, Focused crawler that listens to the user search queries. When a search query is given to the crawler, it will start crawling from authentic page and if the page is relevant, the page is searched for the related keywords/synonyms of the given keyword. This helps to check the authenticity of the page. Then the links are send to the content adaptation to process further. Third, update crawler will update the information stored in the storage server in a periodic manner. Thus Focused crawler to keep the crawl more up-to-date.

3. Content adaptation

The main purpose of content adaptation is to filter through the contents of the web pages to avoid the unnecessary details like advertisement. It uses Lossy compression. The URLs received from the crawler are downloaded fully and the contents are adapted and stored. When a user requests a web page, its contents should be restructured according to the user preferences and bandwidth. So some content should be stripped out and stored separately. The text data is extracted and stored separately in the storage server for later retrieval. The quality of the images varies from low, medium and high.

Advantages of proposed system:

- An effective web search over low bandwidth networks
- Search can be managed according to the user query
- Provide quickness for the retrieval of documents
- Reduce the greater search time for the user (Time Saving)
- Minimal use of resources

Statement of Scope

The system finds its application in a variety of fields.

- In research areas where specific topic specialization is done
- To provide a more user friendly design to an internet searching
- To provide quickness for the retrieval of document
- Improved web search within minimal network connectivity
- Provides fact based data

System Context

This system is used to provide a financial benefit and for quickness. The system makes the search more efficient in low bandwidth networks. The main objective is to search details from internet according to the topic given by the user. It supports an expanded search query interface which allows a user to specify additional query terms to maximize the utility of the results returned by a search query. It performs several optimizations to pre fetch pages to best satisfy a user based on their search preferences.

The objective is to develop a new system which will increase the speed of existing system by reducing complexities. It is a system which is intended to support effective web search over low bandwidth networks.

The Information Retrieval System is used:

- To provide a more effective web search over low bandwidth networks.
- To provide a more user friendly design to an internet searching.
- To provide quickness for the retrieval of documents.
- To reduce the greater search time for the user.

Subsystem overview

Each unit in the system is described here point wise.

1. Local Search

The local search is a searching process within the storage server which contains the downloaded pages.

The functionalities are

- Indexing
- Search using keywords/links
- Present the search result in a structural way

When the user enters the search query in the text field available, the system performs a storage server lookup to find matching keyword .This is done with the help of a database table keyword_store. The list of the keyword present in the storage server is listed in this table. If the query string is in the storage server, the relevant contents are retrieved to the user. If the query string is not in the storage server, then the system performs internet search. I.e. The system searches the query string in the web and retrieves the relevant pages. Then the content of the page is adapted and stored in the storage server. The next time,

when the user searches for the same query string, the contents are retrieved from the storage server.

2. Focused Crawling

It is the process of crawling for relevant pages in the internet based on a given topic. It includes the process of filtering for finding authentic pages. It starts crawling from an authentic page. It employs Breadth First Search (BFS) to find the links. Among the available links, more relevant links are retrieved. This module also performs the updation process of the storage server periodically based on the preset condition.

The steps included are

- Start crawling from a known page
- Based on the context, links are selected and cached
- The process is repeated for the links present in the page
- Updation of the storage server periodically
- Automatic crawling for all available links continuously

3. Content Adaptation

The main purpose of content adaptation is to filter the contents of the web pages to save the server page and to make the data more abstract. When a user requests a web page, its contents should be restructured according to the user preferences and bandwidth. So some content should be stripped out and stored separately. The text data is extracted and stored separately. The type of lossy compression is performed on the content to adapt the content. The processes included are:

- Classification
- Filtering
- Selection
- Compression

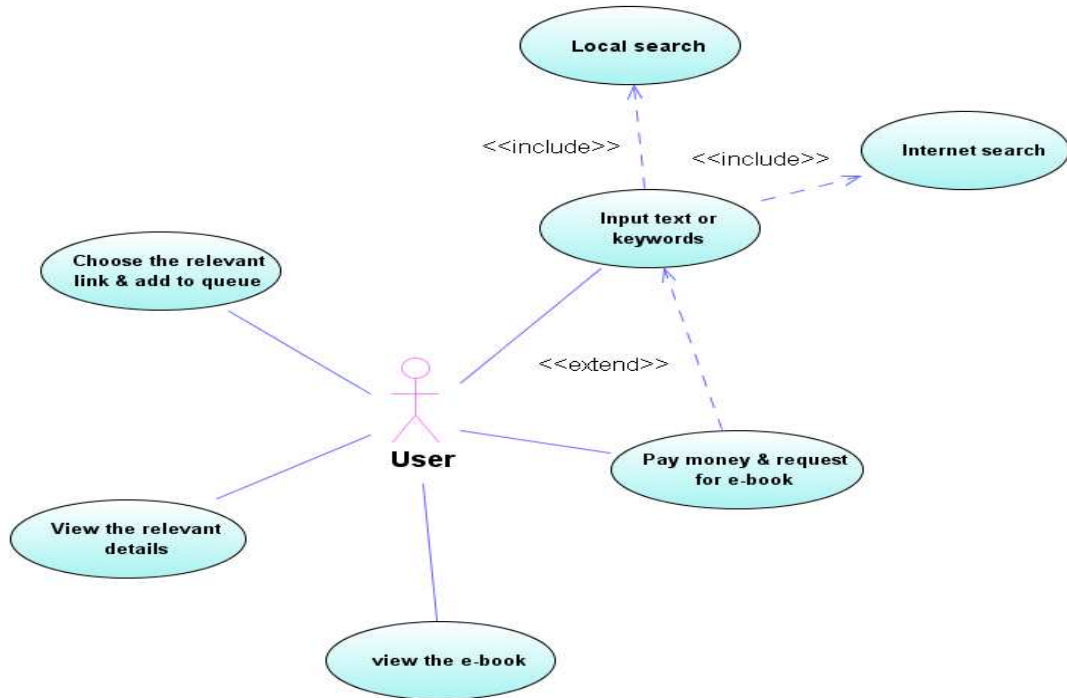
Usage Scenario

This section provides a usage scenario for the software. It organized information collected during requirements elicitation into use-cases.

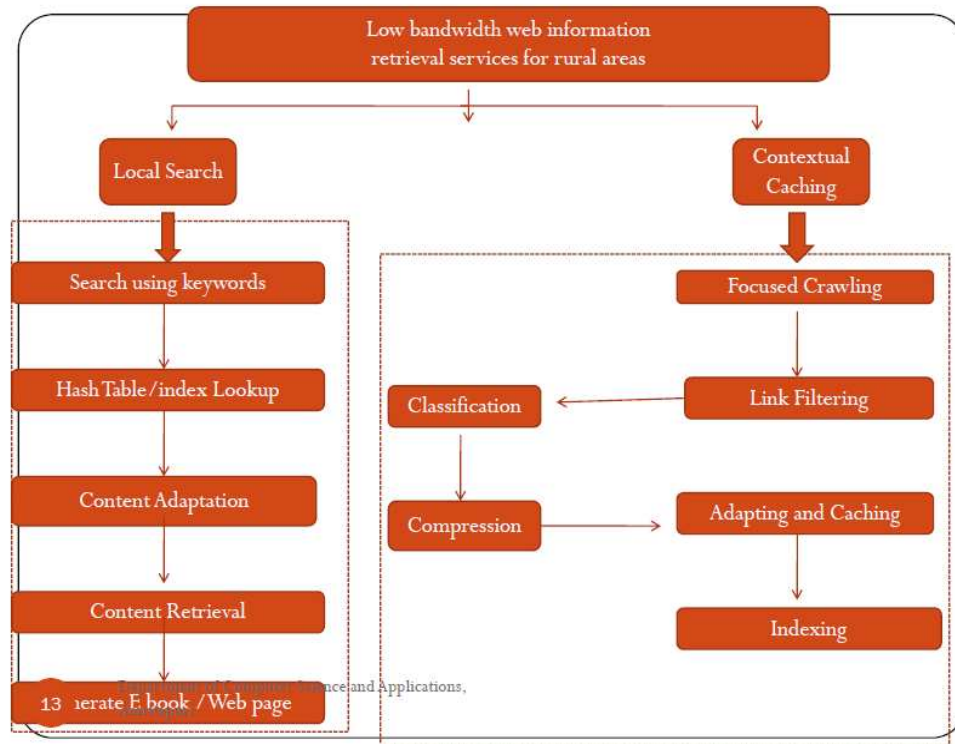
User profiles

The user of this system is the web user who is trying to get the output for the searched query.

Use-case



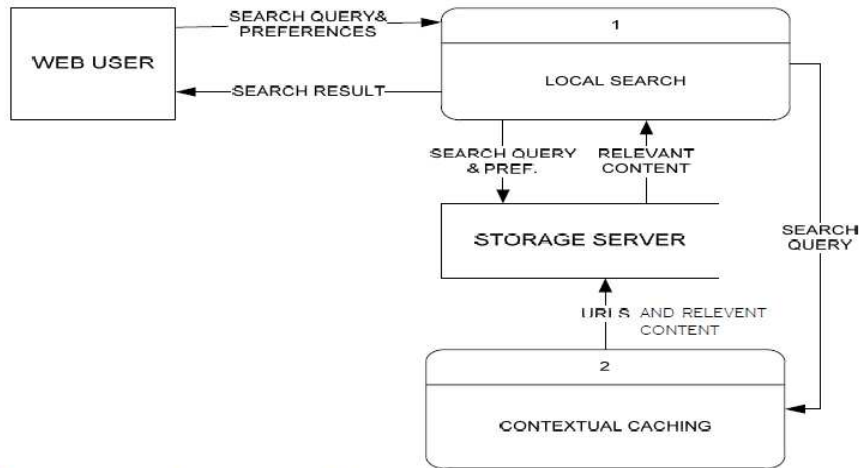
System Design



DATA FLOW DIAGRAMS

Data flow model for this system is developed using Gane-Sarson methodology. The following data flow diagrams (DFDs) represent the movement of data within the system. They concentrate less on the actual functions and data constructs of programmers and more on the general processes inherent to the overall system. We started at the top of the system and moved deeper into the processes. When examining an existing information system or analyzing the information that is going to be designed, it is important to recognize what the data is, where the data comes from, how it passes from one point to another within the information system, and how it will be used by the intended audience or user. The amount of detail specified in this document will include a level two representation for most functions and a level three where necessary. All diagrams include references to additional levels when applicable. Expanded functions are referenced using numbered tabs, which provide the corresponding diagram number.

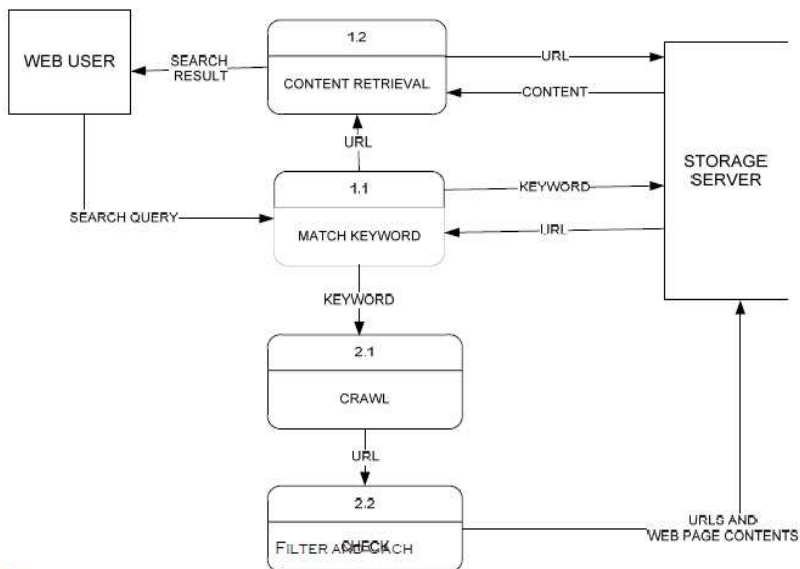
DFD - Level 1



14

Department of Computer Science and Applications,
Amritapuri

DFD - Level 2



15

Department of Computer Science and Applications,
Amritapuri

DATA DICTIONARY

The table used in this system is:

keyword_store

The keyword_store table contains the fields

key_id,

original_keyword,

mod_keyword

d_date.

Keyword_store stores the information about a keyword present in the storage server. The

key_id is the system generated id for each keyword that is searched in the system.

original_keyword denotes the keyword in its original format as specified by the

user.mod_keyword is the format of the keyword that is modified by the system. d_date this

field denotes the date in which the keyword is entered in the table, which is the date in which

the keyword is searched. The updation of the storage server is done based on the d_date of a

keyword.

Field Name	Data Type	Attributes	Description
key_id	Number	Primary Key	ID of the keyword
original_keyword	Text	Not Null	Original keyword

mod_keyword	Text	Not Null	Modified keyword
d_date	Date	Not Null	Date of download

Functional Description

Entering the search query

The user of the system can enter the search query along with the preferences. He can specify the type of search he wish to perform like URL Search or a data search through the interface .The user has option to select the bandwidth type as low, medium or high according to which the contents are retrieved for the user by the system. And also he can select the type of search he wants to perform as deep, broad or normal. Based on these selections the search is performed.

Direct Search

Direct search is the functionality in the system in which the user can directly specify the url request in the textbox and the page corresponding to the request is retrieved and presented to the user. This functionality provides direct access to the page.

Local Search

In order to be utilized in low bandwidth networks the system first performs the search for the data in the storage server with the help of a database table Keyword_store which containing the details of already searched keywords. If the data related to the user specified keyword is present, it is retrieved and presented to the user from the storage server.

If not, the system will invoke the crawler part to perform the web search for the data. This local search mechanism is time saving and efficient.

Content Retrieval

Content retrieval deals with presenting the data according to the user preferences. The data is presented to the user based on the type of search, bandwidth of the network and the request type. Based on the bandwidth type the data is presented to the user as for low bandwidth only text, for medium bandwidth text and images, for high bandwidth text and high quality image is delivered to the user.

Web Crawling

Crawling is the process performed to search for the keyword in the web. This process is invoked only if the data is not present in the storage server. Crawling performed is a type of focused crawling in which the data retrieved is relevant and authentic. The crawling starts from an authentic page. The breadth first search mechanism is used. All the links of the pages are retrieved and checked whether they are relevant or not. If a page is found to be relevant the links are retrieved and presented to the content adaptation module. The updation of the storage server is done by crawling.

Checking for the relevant pages

While retrieving the links the pages are checked for relevancy based on the keywords and related keywords. The page that contains maximum number of keywords and related keywords are considered to be relevant. The links of such pages are retrieved and given directly to the content adaptation module. The authenticity of a page achieved by starting crawling from an authentic link.

Downloading

The module downloading includes the process like downloading the content of the relevant link given by the crawler. The downloaded content is subjected adaptation.

Content Adaptation

Content adaptation is the process of formatting the data. The main purpose of content adaptation is to filter the contents of the web pages. When a request elicit mass amount of data, it will cause the server space. So some content should be stripped out and stored separately. The type of compression we use is the lossy compression The main tasks are classification, filtering, selection and compression. First we will separate the contents according to the tags and based on the user request, irrelevant tags are commented and the web page is cached. Textual data store separately. This data is common for all type of users. Images are stored separately in different category. I.e., same picture is stored in different bandwidth category with different size. Other scripts are stored in higher bandwidth category.

CODING

Modules

I Local Search

- Database lookup
- Content retrieval

II . Contextual Caching

- Focused crawling
- Link filtering and publishing
- Classification
- Compression / Content adaptation
- Caching and Indexing

- **Algorithm –**

Input: keyword from user

Output: retrieved Page to the browser / or to the local store

BEGIN

Input Keyword from the user

If(Input present in the local database of keywords)

 localsearch()

Else

{

 crawl();

 publishurl();

 contentadapt();

 localstore();

}

END

- **Algorithm – Local Search**

Input: keyword from user

Output: retrieved Page to the browser / or to the local store

BEGIN

Input Keyword from the user

If(Input present in the local database of keywords)

{

 read user preference;

 retrieve link from local store based on the preference;

 display result in browser;

}

Else

```
{  
    contextual caching();  
    display result in browser;  
}
```

END

- **Algorithm – Contextual Caching**

Input: keyword from localsearch()

Output: content stored in local store

BEGIN

Input = keyword from localsearch();

urllist = Crawl(input);

Publish(urllist);

Contentadapt();

Localstore();

END;

Screen shots



[\[REFRESH\]](#)

#	Query/Page Request	Status/ETA	Options
1.	root.text.com/4guysfromrolla/aspnet/articles/012203-1.2.aspx	Unknown	[REMOVE]
2.	root.text.com/devdirect/ALL/Imagingmanipulation_PCAT_1862.aspx	Unknown	[REMOVE]
3.	root.text.com/vwd-cms/open-source/source-code-viewer.aspx	Unknown	[REMOVE]
4.	root.text.com/15seconds/issue/050526.htm	Unknown	[REMOVE]

[\[REFRESH\]](#)



[ASPFAQs.com](#)
[Message Board](#)
[Related Web Technologies](#)
[User Tips!](#)
[Coding Tips](#)
[Search](#)

Sections:
[Book Reviews](#)
[Sample Chapters](#)
[Commonly Asked](#)
[Message Board](#)
[Questions](#)
[Headlines from](#)
[ASPWire.com](#)

True Image Resizing, Part 2

By *Scott Mitchell*

• [Read Part 1](#)

In [Part 1](#) we examined how to create an ASP.NET Web page, *ShowImage.aspx*, that would display an image. This ASP.NET Web page could then be referenced via an HTML `` tag. In this part we'll look at how to extend the *ShowImage.aspx* so that it can produce thumbnail images; then, we'll see how to tie this lesson together with the lessons learned from the article [Displaying a List of Scaled Images](#).

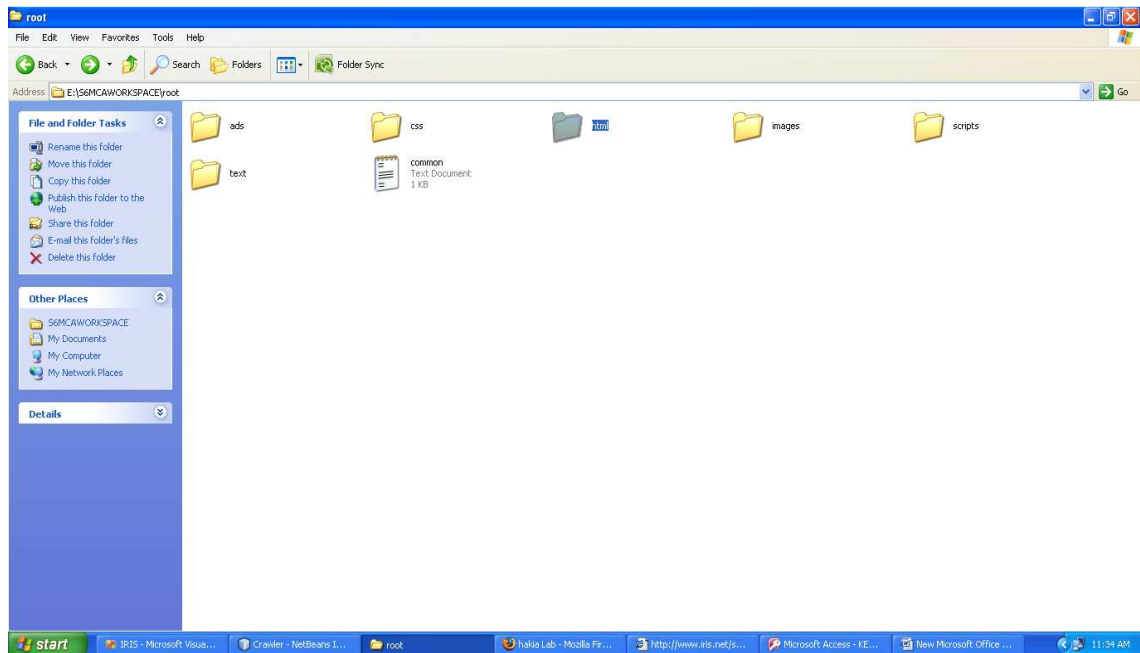
[-continued-](#)

Altering ShowImage.aspx to Create a Thumbnail
Earlier we saw that we could create an ASP.NET Web page called *ShowImage.aspx* that could be passed in an image file name and display the image. In addition to accepting the file name, let's augment this ASP.NET Web page so that we can pass in an optional height and width. When a height and width are passed in, the image displayed is first dynamically resized to the specified height and width. The following code implements the needed changes in *ShowImage.aspx*.

```
<<Import Namespace="System.Drawing.Imaging" *>  
<script language="VB" runat="server">  
Function ThumbnailCallback() as Boolean  
Return False
```

The screenshot shows the NetBeans IDE 6.1 interface. The Output window is active, displaying the following text:

```
init:  
deps-jar:  
compile:  
run:  
Crawler Initialized...  
Crawler is ready...  
http://en.wikipedia.org/wiki/Gandhi  
downloadPage : java.io.IOException: Server returned HTTP response code: 403 for URL: http://en.wikipedia.org/w/index.php?title=Gandhi&redirect=no  
http://en.wikipedia.org/wiki/Gandhi_(disambiguation)  
http://en.wikipedia.org/wiki/Assassination_of_Mohandas_Karamchand_Gandhi  
http://en.wikipedia.org/wiki/Marital_Gandhi  
http://en.wikipedia.org/wiki/Mandira_Gandhi  
http://en.wikipedia.org/wiki/Gujarati_language  
http://en.wikipedia.org/wiki/Satyagraha  
http://en.wikipedia.org/wiki/Civil_disobedience  
http://en.wikipedia.org/wiki/International_Day_of_Non-Violence  
http://en.wikipedia.org/wiki/Swaraj  
http://en.wikipedia.org/wiki/Sabarnati_Ashram
```



TESTING

Types used

- **Unit testing**
- **Integration Testing**

Test case: 1

Test case Name: Keyword

Test Description: The keyword is searched in the keyword_store table in the storage server
.If the keyword is not present, the keyword is passed to the crawler.

Expected Result: Relevant links

Actual Result: Relevant links

Pass/Fail: Pass

Test case: 2

Test case Name: Existing Keyword

Test Description: The keyword is search in the keyword_store table in the storage server.
The keyword is present in the server and the relevant pages are retrieved.

Expected Result: Relevant links

Actual Result: Retrieve relevant pages.

Pass/Fail: Pass

Test case: 3

Test case Name: Keyword Messaging

Test Description: The keyword is not present in the storage server, the keyword is passed to
the crawler.

Expected Result: Proper communication between the modules

Actual Result: Crawler receives the keyword.

Pass/Fail: Pass

Test case: 4

Test case Name: Crawling

Test Description: Crawler receives the keyword and starts crawling from an authentic page.

Expected Result: Relevant links

Actual Result: Downloading the pages

Pass/Fail: Pass

Test case: 5

Test case Name: Relevancy

Test Description: The relevancy of the links is checked using the related keywords from WordNet dictionary.

Expected Result: Query Expansion

Actual Result: Relevant links

Pass/Fail: Pass

Test case: 6

Test case Name: Link Messaging

Test Description: The relevant links retrieved by the crawler given to the content adaptation module which performs the lossy compression of the web page.

Expected Result: Proper communication between the modules

Actual Result: Receive the links

Pass/Fail: Pass

Test case: 7

Test case Name: Content Adaptation

Test Description: Downloading the data corresponding to relevant links and performing the lossy compression and the data is stored separately.

Expected Result: Downloading

Actual Result: Compressed data stored separately.

Pass/Fail: Pass

Test case: 8

Test case Name: Indexing

Test Description: Downloaded pages are indexed

Expected Result: Proper indexing

Actual Result: Indexed pages

Pass/Fail: Pass

Test case: 9

Test case Name: Adding keyword

Test Description: Insert keywords in to the table.

Expected Result: Keyword storing

Actual Result: keyword added

Pass/Fail: Pass

CONCLUSION

Within the limited period of time we conducted feasibility study, detailed system study and tracked several implementation issues. The current system is a new proposal for efficient and relevant searching over low bandwidth networks. This project was not done simply as part of the course work. This is an ongoing topic under a research team of Amrita E learning Lab. Due to the lack of time, advanced features like semantic search and e-book generation remains beyond the scope of this Record.

BIBLIOGRAPHY

<http://cater.cs.nyu.edu/wiki/index.php/RuralCafe>

http://www.googleguide.com/google_works.htm

<http://www.google.com/complete/search>

<http://www.en.wikipedia.org>

<http://wordnet.princeton.edu/>

<http://activemq.apache.org/>

<http://www.hakia.com/>

References (Related works identified)

· Web Search: Rural Café: - By CATER (Cost effective Appropriate technologies for emerging regions) , New York University

<http://cater.cs.nyu.edu/wiki/index.php/Home>

· Content adaptation: Loband - Loband simplifies web pages by filtering out images and other bandwidth-intensive content so that they can be received faster over slow Internet connections.

www.loband.org .

· Local Content Search: Apache Lucene

<http://lucene.apache.org/java/docs/>

FUTURE ENHANCEMENT

The future enhancements of the system are the advanced features like

- Semantic search
- E-book creation

APPENDIX

Tools used:**.NET:**

.Net is a “Software platform”. It is a language-neutral environment for writing programs that can easily and securely interoperate. Rather than targeting a particular hardware/operating system combination, programs will instead target “.NET”, and will run wherever .NET is implemented. NET is also the collective name given to various bits of software built upon the .NET platform. The components that make up .NET -the platform are called the .NET Framework.

The .NET framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet. The .NET framework is designed to fulfill the following objectives

- To provide a consistent object-oriented programming environment whether object code is stored and executed locally, but Internet-distributed or executed remotely.
- To provide a code-execution environment that minimizes software deployment and versioning conflicts.
- To provide a code-execution environment that guarantees safe execution of code including code created by an unknown or semi-trusted third party.
- To provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.
- To make the developer experience consistent across widely varying types of applications, such as Windows-based application and web-based applications.
- To build all communication on industry standards to ensure that code based on the .Net framework can integrate with any other code.

The .NET framework has two main components:

1. The Common Language Runtime
2. A hierarchical set of Class Libraries

Visual Studio.NET:

Visual Studio .NET is a complete set of development tools for building ASP Web applications, XML Web services, desktop applications, and mobile applications. Visual Basic .NET, Visual C++ .NET, and ASP.NET using C# all use the same Integrated Development Environment (IDE), which allows them to share tools and facilitates in the creation of mixed-language solutions. In addition, these languages leverage the functionality of the .NET Framework, which provides access to key technologies that simplify the development of ASP Web applications and XML Web services.

NetBeans Platform

The **NetBeans Platform** is a reusable framework for simplifying the development of other desktop applications. NetBeans refers to both a platform for the development of applications for the network (using Java, JavaScript, PHP, Python, Ruby, Groovy, C, and C++), and an integrated development environment (IDE) developed using the NetBeans Platform.

When an application based on the NetBeans Platform is run, the platform's Main class is executed. Available modules are located, placed in an in-memory registry, and the modules' startup tasks are executed. Generally, a module's code is loaded into memory only as it is needed.

Applications can install modules dynamically. Any application can include the Update Center module to allow users of the application to download digitally-signed upgrades and new features directly into the running application. Reinstalling an upgrade or a new release does not force users to download the entire application again.

The platform offers services common to desktop applications, allowing developers to focus on the logic specific to their application. Among the features of the platform are:

- User interface management (e.g. menus and toolbars)
- User settings management
- Storage management (saving and loading any kind of data)
- Window management
- Wizard framework (supports step-by-step dialogs)

Apache ActiveMQ

Apache ActiveMQ is an open source (Apache 2.0 licensed) message broker which fully implements the Java Message Service 1.1 (JMS). It provides "Enterprise Features"^[1] like clustering, multiple message stores, and availability to use any DB as a JMS persistence provider besides VM, cache, and journal persistency.

Apache ActiveMQ NMS

Messaging is a method of communication between software components or applications. A messaging system is a peer-to-peer facility: A messaging client can send messages to, and receive messages from, any other client. Each client connects to a messaging agent that provides facilities for creating, sending, receiving, and reading messages.

Messaging enables distributed communication that is *loosely coupled*. A component sends a message to a destination, and the recipient can retrieve the message from the destination. However, the sender and the receiver do not have to be available at the same time in order to communicate. In fact, the sender does not need to know anything about the receiver; nor does the receiver need to know anything about the sender. The sender and the receiver need to know only what message format and what destination to use.

The NMS API (*.Net Message Service API*) provides a standard C# interface to Messaging Systems. There could be multiple implementations to different providers (including MSMQ).

NMS API currently supports all of the features of JMS in a simple pure C# API and implementation apart from XA. Current features include

- creating & disposing of connections, sessions, producers, consumers
- sending of messages to topics, queues with durable or non durable along with temporary destination support
- synchronous consuming (blocking receive, receive with no wait or receive with a timeout)
- asynchronous consuming (adding a Message Listener to be dispatched in the thread pool)
- JMS message header support along with custom properties
- Text, Bytes and Map message support
- support for JMS transactions (sending and acknowledging multiple messages in an atomic transaction)
- redelivery of messages in rollbacks up to some configured maximum redelivery count

Lucene

Apache Lucene is a high-performance, full-featured text search engine library written entirely in Java. It is a technology suitable for nearly any application that requires full-text search, especially cross-platform. While suitable for any application which requires full text indexing and searching capability, Lucene has been widely recognized for its utility in the implementation of Internet search engines and local, single-site searching.

WordNet 2.0

WordNet is a large lexical database of English, developed under the direction of George A. Miller. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. The resulting network of meaningfully related words and concepts can be navigated with the browser. WordNet's structure makes it a useful tool for computational linguistics and natural language processing.