WEB SITE ANALYZER: IMPLEMENTATION

YUKUI LU

A MAJOR REPORT

IN

THE DEPARTMENT

OF

COMPUTER SCIENCE

PRESENTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF MASTER OF COMPUTER SCIENCE

CONCORDIA UNIVERSITY

MONTREAL, QUEBEC, CANADA

AUGUST 2003

Your file Votre référence
ISBN: 0-612-83915-X
Our file Notre référence
ISBN: 0-612-83915-X

Abstract

Web Site Analyzer

Yukui Lu

This report presents a Web Site Analyzer tool (WSA) which analyzes the structure of a web site by typing a URL name. There are five categories of links presented: Links Reference To, Links Referenced By, Leaf Links, Dead Links and Foreign Links. The results are presented in a folder of a hierarchy tree menu, which allow the user to easily span and close the folder and to navigate the links of each category. The results are showing as well the relationships between links. The architecture of this application is a typical three-tier Web application. The software is implemented in JSP, Java, Java Bean, Java Script. Apache Tomcat is used as a web server, and Access database is used as a demo database to store the links of search results.

This application demonstrates a dynamic, no size limitation, easy to use, professional user interface tool of Web Site Analyzer.

Acknowledgements

I would like to take this opportunity to thank my supervisor Dr. Peter Grogono for his professional supervision and intellectual guidance. Thanks to Dr. Harutyunuan, the examiner of this work, and Bogdana Marinescu, my colleague in Motorola, for their valuable comments on this report. Thanks also to Halina Monkiewicz, the Graduate Program Advisor, for her timely encouragements and suggestions during my master program study.

I would also like to express my great thanks to my husband, Jianzhong, my mother-in-law, my daughter, Hanglu, and my son, Brian for their love and support. The completion of this major report would not have been possible without them.

Table of Contents

1 Introduction

More and more enterprises and people have their own web sites nowadays, and these web sites are getting larger and more complex, even a personal web site may contain thousands of files. How to maintain these web sites has become a significant task for web masters. The web masters or developers may want to know the structure of their web sites, the number of files and the relationship between these files in their web sites. They may also concern if there any dead links in their web sites that prevent the end users from accessing. Though there are tools available to help, they are slow due to remote access, expensive, or cumbersome.

This report presents the implementations of an application named WSA (Web Site Analyzer). The WSA system analyzes a web site structure and generates a report as a web browser user graphic interface that presents the categories of the files in this web site. The category is based on the web accessibility. The main purpose of this application is to let the web masters or personal web site owners to easily know and maintain the file accessibility structure of their web sites.

In this document, section 1 describes the introduction and milestones. Section 2 describes the original requirements and the analysis of the requirements. Section 3 introduces some technologies and tools that were applied to this project such as JavaServer Page, Servlet, Tomcat, Treeview and the reason why they were chosen. Section 4 presents the three-tier Web based application architecture of this project. Section 5 discusses the main design considerations, the user interface design, and the interaction between different tiers. Section 6 explains the implementations and functionalities of JSP pages and Java classes, as well as the database structure used in this project. Section 7 describes the system test cases and the experimental results, as well as how to analyze a web site by using this application. Section 8 introduces and analyzes the advantages and disadvantages of the previous version of WSA and other related Web Site analyzer tools by comparing with this new WSA tool. Section 9 makes a conclusion for this research work and provide recommendations for future works. Appendix--WSA Installation Guideline provides the detailed steps to setup the environment and install WSA tool.

1.1 Definitions and Acronyms

WSA	Website Analyzer
URL	Uniform Resource Locator
JSP	JavaServer Page
ASP	Active Server Page
HTML	Hypertext Markup Language
PHP	PHP: Hypertext Preprocessor
TreeView	A tool of cross-browser dynamic DHTML tree builder.
Servlet	A small program that runs on a server.
Tomcat	A servlet container with JSP environment
JDBC	Java Database Connectivity
ODBC	Open Database Connectivity
JDBC-ODBC	Implements JDBC operations by translating them into ODBC operations.
JDK	Java Development Kit

SSL	Secure Sockets Layer
HTTP	HyperText Transfer Protocol

1.2 Scope and Project milestones

1.2.1 Scope

The scope of this the project is to develop a tool that analyzes the file accessibility of a web site and give the user a graphic report of analysis. The user starts WSA by giving the URL address as input via the user interface. The program scans HTML files recursively and builds a tree menu graph.

1.2.2 Joint Effort

This project is the joint-effort from two developers (Fan Jiang and Yukui Lu) who are working as a pair through the every phase of this application. We took the advantage of co-operation between two persons by pair analyzing, pair design, pair programming and pair testing. This pair working style is encouraged by the extreme programming methodology, which we believe in. Therefore, our major report documentation may share some common works, mainly in the Introduction, Requirements, Test, Related Works and Conclusion sections. According to the architecture design, there are two components with nearly no dependencies. Therefore, each of us leads the effort of one component, even though we still work as a pair from time to time. The following sections of this document are written separately: Architecture, Design, Implementation, Technology

Tools (only presented in Yukui Lu's report), XP Programming Methodology (only presented in Fan Jiang's report).

1.3 Milestones

This project started in January 2002 and Professor Peter Grogono was the supervisor of this project. It did not start from scratch, it was based on the WSA C++ version implemented by Professor Peter Grogono and the Java version implemented by the Graduate student Yuan Xu [18] which will be called old WSA below. The parse algorithms and code were reused from the previous versions of implementation.

The milestone and procedure of this work are the following:

1. Be familiar with the topic and related works in this field as well as to be familiar with the old WSA tool.
2. Analyze the architecture, design, implementation, and performance of the old WSA.
3. New WSA architecture and design, background technologies and tools learning.
4. Implementation—Coding and Unit Testing.
5. Test cases design and System Testing
6. Documentation
7. Make a conclusion for this research work and provide recommendations for future works.

2 Requirements

This section describes the original requirements and the analysis of the requirements.

2.1 Original Requirement

The original requirements for this project covers the requirements given by Professor Peter Grogono. [1]

The Analyzer should find all files in the starting directory and all files that are:

1. Reachable from those files by traversing links and
2. On the local system.

The second condition is intended to prevent the Analyzer returning the entire World Wide Web as its result.

The Analyzer reports:

1. The files that it found;
2. For each file, the links from it and the links to it;
3. "Orphans" ---- files with no incoming links;
4. "Leaves" ----files with no outgoing links;
5. "Foreigners"---- names of remote files.
6. "Dead link" – URLs which are not found.

In order to work properly, the Analyzer has to parse HTML, at least partially. As a side effect of its work, it could produce a list of warnings of HTML errors.

2.2 Requirements Analysis

2.2.1 Links

Based on 2.1, the links are categorized to following five types of links:

1. Links Reference To (Incoming links to the specific file)
2. Links Referenced By (Outgoing links from the specific file)
3. Foreign Links
4. Dead Links (All un-reachable links, ex. permission denied and inaccessible)
5. Leaf Links

There is some relationship among these types of links. For example, leaf links, and foreign links may have incoming links.

In this project, the orphan link was not implemented, since the traverse algorithm cannot reach this type of links. This will be mentioned on the future work section.

2.2.2 GUI

1. The Project requires that the user to have a graphic interface to input the base URL to be analyzed.
2. The Application should have the clear visual presentation showing the structure of the analyzed web site.

2.3 Development Environment

The development environment and software tools that were used to develop WSA are described below:

- Platform – WINDOWS 2000
- Developing languages – JAVA, JSP, JavaScript, HTML
- Database – Microsoft Access 2000
- Web Server – Apache Tomcat 3.2.4
- IDE – JBuilder Foundation 4.0.

3 Technical Tools

This project is a three-tier web-based application. Some tools and technologies have been used in this project, such as JavaServer Page, Servlet, Tomcat, Treeview, JavaBean, JDBC, etc. In order to have a better understanding of the contents described in this paper, a brief introduction of some of the tools and technologies used are presented.

3.1 JSP

JavaServer Page, or JSP in short, is a Java-based technology that helps to simplify the process of developing dynamic web sites. JSP specification is developed by Sun Microsystems under the Java Community Process (JCP). It is the product of industry-wide collaboration with industry leaders in the enterprise software and tools markets. [3]

JSP technology enables to mix regular, static HTML with dynamically generated content from servlets [2]. It is a type of server-side scripting language. JSP offers web designers and developers the possibility to quickly incorporate dynamic elements into web pages using embedded Java and a few simple markup tags. JSP files contain traditional HTML along with embedded code that allows the page designer to access data from Java code running on the server. [4]

The difference between traditional HTML and JSP are described following. For HTML pages, all the requested documents are static files; it means that the documents requested never change. No matter who requests them, when they are requested, or if additional parameters are included with the request, the server just goes to find the same target file and returns it. However, more and more dynamic data is delivered over the web, such as up-to-the-minute stock prices, online shopping, personal email messages, etc. The old HTML pages could not satisfy these dynamic requests. The web server has to do some additional data processing in order to generate customized response to dynamic request.

JSP comes with such dynamic content technologies. With JSP, when a user requests the page, the code portions of the page are executed at the time the request is received, and the dynamic content generated by this code is compound into the page before it is sent to the user.

There are also other different technologies that can archive this goal, such as ASP, PHP, etc. The biggest advantage of JSP is its Java-based technology. Versus to ASP which from Microsoft, the dynamic part of JSP is written in Java, not VBScript, it is more power and better switch to complex application that require reusable components. Second, JSP is portable to other OS and web server and not locked into windows NT/2000 and IIS while ASP does. Versus to PHP which is HTML-embedded scripting language, JSP is easy to learn and use. Since JSP is written in JAVA that already has an extensive API for networking, database access, distributed objects, while PHP requires learning an entirely new language. [2]

By using JSP, web developers are able to maintain and change their web pages easier. Because JSP separate presentation from implementation logic by combining standard markup text with scripting elements and object-oriented components, it allows designers to change the overall page layout without changing the underlying dynamic content. [3] JSP provides an excellent front-end technology for applications that are deployed over the web.

3.2 Servlet and Tomcat

Servlets are programs that run on the web server, acting as a middle layer between a request coming from a web browser or other HTTP client and database or applications on the HTTP server [5]. One can think of that servlet as an applet that runs on the server side without a user interface.

Servlets are not tied to a specified protocol; it can use many kinds of client-server protocols. But when we talk about Servlets, we are usually meaning HTTP Servlets because of the popularity of the HTTP.

There are three typical uses for HTTP Servlets:

The first one is to process and store data submitted by an HTML form. The second one is to provide dynamic content, such as return the results of a database query to the client. Another use is to manage the state information on top of the stateless HTTP, such as for an online banking system that manages transactions for many concurrent customers and maps every request to the right customer.

Servlets, being Java Application, have direct access to the full range of Java functions, asuch threading, network access, and database connectivity [6]. It is widely used to build interactive Web applications. The most popular HTTP Web Servers on the Internet are Apache, Microsoft-IIS, Zeus, and SunONE. [9] For non-commercial development and testing purpose, Apache Tomcat is the best choice. Tomcat is a very popular, free, and small Web server which can be installed on the desktop machine and used as a small stand-alone server, or as an add-on be integrated into a existing server, such as Apache, IIS, and Netscape web servers.

Tomcat is the official reference implementation of Java servlet and JavaServer Pages technologies [7]. It provides Web developers with a simple, component-based, platform-independent mechanism to build Web-based applications. Tomcat is a servlet container for running servlets and JSP, as well as accessing to all Java APIs, including the JDBC API to access enterprise databases. For many web applications this is sufficient to provide the functionality to solve the problem elegantly and rapidly. [8]

3.3 JDBC and JDBC-ODBC Bridge

To access any database, a database access driver is requested. JDBC technology is introduced by Sun Microsystems. It is an application programming interface between Java programs and database management system. Like Microsoft's ODBC, JDBC is a call-level interface which means a programs uses method or function calls to access its interface [6]

JDBC provides a standard interface to all database management system, JDBC queries for different database require few or no changes. [6] It allows the access to any virtually and tabular data source from the Java programming language. With JDBC, developers are able to easily execute common SQL statements, such as create tables, insert values into them, query the tables, retrieve the results of the queries, and update the tables, etc. Additional, the JDBC API offers the advantage of JavaBeans technology and Enterprise JavaBeans technology, one of it is "Write Once, Run Anywhere™" capabilities.

Though JDBC are able to connect to most of the databases, some additional technologies are required. Usually, there are two popular database access drivers are widely used in today's APIs: ODBC and JDBC. Microsoft uses ODBC (Open Database Connectivity) drivers with most of their software while Java uses JDBC (Java Database Connectivity) drivers to connect to various databases. JDBC cannot directly connect to an ODBC database. In order to access an ODBC database with JDBC, in our case is Microsoft Access 2000, then, JDBC-ODBC Bridge driver comes in and overcomes this gap.

The JDBC-ODBC Bridge is a JDBC driver that implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge implements JDBC for any database for which an ODBC driver is available. The Bridge is implemented as the sun.jdbc.odbc Java package and contains a native library used to access ODBC. [15]

3.4 Treeview

Treeview, a JavaScript Applet, is a cross-browser dynamic HTML tree with online visual builder. [10] It is a free software tool that authored by *Marcelino Martins*. By using this tool, web developers are able to easily and quickly build their own web pages with tree menus by modifying and customizing some codes following certain rules, while sharing the same general tree structure.

With Treeview, a tree menu can be placed on the frame of a frameset or on a regular frameless page. With a tree menu control, the information is displayed in a hierarchical order, with the home topic at the top as folders and the subordinated items or "documents" underneath. Because its hierarchical structure can hold a very long number of links, while still only showing a small number of them at a time, it is very effective used as main navigation index. Clicking on the minus (-) or plus (+) sign will expand or collapse the tree immediately on the browser. Clicking on a link will load a new page on the right pane or on a new window. [10]

The latest version of Treeview 4.3 supports all major browsers and versions, such as IE4.0+, NS4.7+, Mozilla 0.9+, Opera 7.0+. Its look and feel flexibility allows developer to customizable folder and document icons, option to wrap node text into multiple lines; Its fast performance allows user to load unlimited hierarchical levels and thousands of nodes in tree menu in a short period of time. Its optional server integration advantage allows developers to extend server-side browsing of files and directories, server-side connection to databases.

4 Architecture and Design

4.1 Architecture

WSA is a three-tier Web based application. The advantage of a multi-tiered applications is these tiers separated the aspects of an enterprise application as well as the business logic so that each tier can change at its own rates while minimizing the change to the entire application. [11] In WSA, it divides the application in three layers: Presentation, Business Logic, and Data layer. Figure 1 illustrates the high level architecture of WSA.

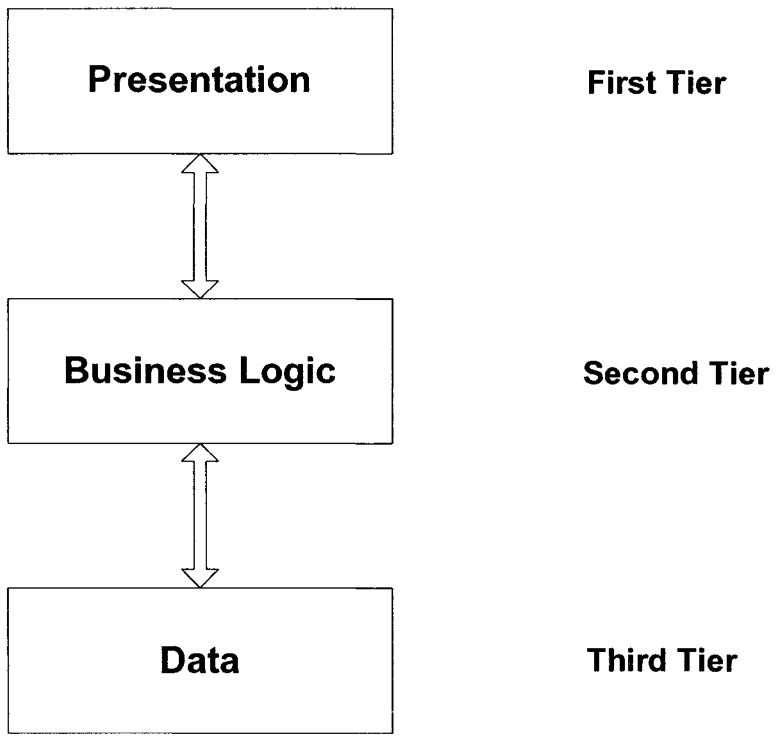


Figure 1 WSA High-level Architecture

Below is a brief explanation of the three tiers.

4.1.1 The Presentation Layer

The first layer Presentation deals with the user input and presents the result, this layer contains all technologies used on the client side.

4.1.2 Application Layer

The second layer Business Logic contains all the code of the server-side technology. This layer contains a web server running scripting language providing database access, pass data to user interface and handle any input from the UI as well.

The application functionality in the Application Layer is a set of JSP pages and Java classes for page management, database management and data validation.

4.1.3 The Database Layer

The third layer is persistent data storage. All the SQL statements and queries needed for storing, retrieving, and filtering data from the database. Specifically, SQL queries are generated for adding, updating and deleting records into designated tables as well as to provide a tree menu style reporting view.

5 Detailed Design

This section explains the detailed design of WSA.

5.1 System Overview

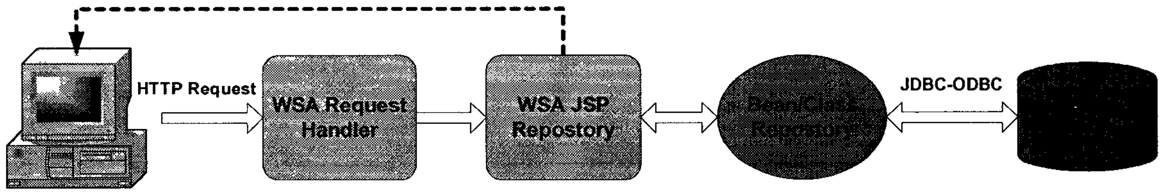


Figure 2. WSA System Overview

Figure 2 shows the System overview of WSA. As a web-based application, WSA implemented a typical JSP three-tier request model. When a request is sent directly to a JSP page, the JSP code controls interactions with JavaBeans components for business and data logic processing, and then displays the results in dynamically generated HTML mixed code. Figure 2 illustrates the flow of information in this model. Following are some main considerations of the WSA design:

1. The new WSA implemented to be web application instead of a Java Applet. User input and system out will be in the same web page with frames sets.
2. Treeview tool was introduced to generate a standard and professional user interface of web application. With menu controls as main navigation index, the results are presented in a hierarchical tree menu that includes folders and links underneath. This gives a very clear relationship between links of the given URL; meanwhile, user can easily span and close the folder to navigate the links.

3. JSP was introduced to implement dynamic Web pages, instead of using static HTML pages. The reason of choosing JSP is because it provides us to simplify the process of developing dynamic web sites. Its Java-based technology offers all of the advantages that the Java language provides with respect to development and deployment.

4. Tomcat was used as web servers. The reason we chose Tomcat as web server for WSA because it is a free, small, and stand-alone server that can be can be installed on the desktop machine. It provides Web developers with a simple, component-based, platform-independent mechanism to build Web-based applications. It can access to all Java APIs, including the JDBC API to access enterprise databases. Also, it allows this WSA used in the local network.

5. We chose to implement the business logic that behind the scene. Java Bean was used to optimize the java codes and archive code reuse.

6. WSA solved the problem of website size limitation of old WSA by using ODBC-JDBC technology and use Access database as storage to save the search results. Another advantage of using a database is user can save and copy this database to local disk for future use. The analysis part and presentation part can work independently, that means user can input and analyze a website at a time, then review the results any time he or she wants.

7. Microsoft Access database was used as data storage to sterilize the search results. Because the data needed to store in WSA is quite simple, only the search result links. Moorcroft Access is a good choice because of its easy to use, easy to

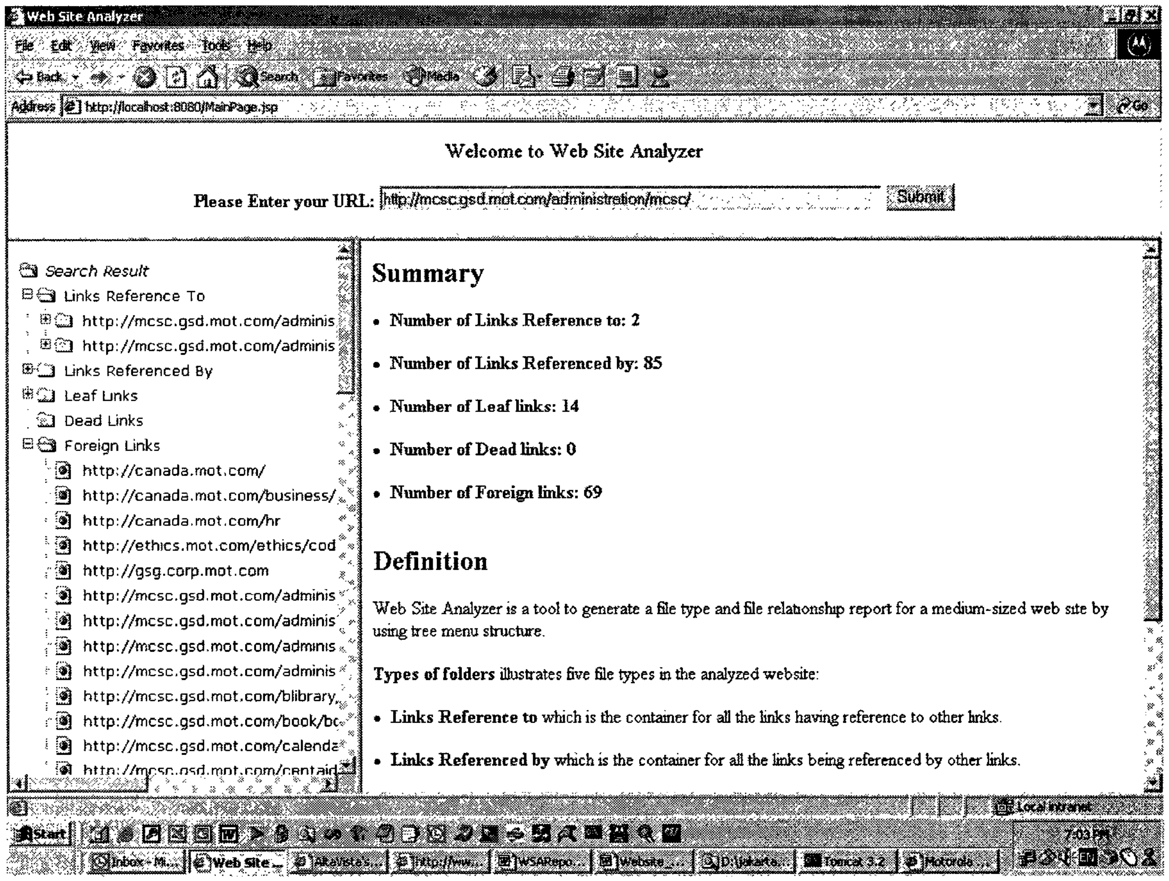
maintain. There is no any extra knowledge needed. Use Access as our database makes difficult database technology simple to accessible.

5.2 User Interface Design

The navigation interface is one of the most important aspects of any application. [10]

One of the main targets of WSA tool is to develop a simple, easy to use and professional user interface. Firstly, to simplify the interface, we decided to use "All in One" interface which means the user input and application outputs are all built in a frameset web page.

To achieve fast performance with standard and professional look and feel, we applied Treeview [10] as a tool to build WSA interface; we placed the tree in web page with frameset to make it have more areas. Figure 3 below shows the WSA user interface.


Web Site Analyzer
File Edit View Favorites Tools Help
Address http://localhost:8080/MainPage.jsp
Go
Local intranet

Welcome to Web Site Analyzer

Please Enter your URL: http://mcsc.gsd.mot.com/administration/mcsc/ Submit

Search Result
- Links Reference To
  - http://mcsc.gsd.mot.com/adminis
  - http://mcsc.gsd.mot.com/adminis
- Links Referenced By
- Leaf Links
- Dead Links
- Foreign Links
  - http://canada.mot.com/
  - http://canada.mot.com/business/
  - http://canada.mot.com/hr
  - http://ethics.mot.com/ethics/cod
  - http://gsg.corp.mot.com
  - http://mcsc.gsd.mot.com/adminis
  - http://mcsc.gsd.mot.com/adminis
  - http://mcsc.gsd.mot.com/adminis
  - http://mcsc.gsd.mot.com/adminis
  - http://mcsc.gsd.mot.com/blibrary
  - http://mcsc.gsd.mot.com/book/bc
  - http://mcsc.gsd.mot.com/calenda
  - http://mcsc.gsd.mot.com/centaid

Summary

- **Number of Links Reference to: 2**
- **Number of Links Referenced by: 85**
- **Number of Leaf links: 14**
- **Number of Dead links: 0**
- **Number of Foreign links: 69**

Definition

Web Site Analyzer is a tool to generate a file type and file relationship report for a medium-sized web site by using tree menu structure.

Types of folders illustrates five file types in the analyzed website:

- **Links Reference to** which is the container for all the links having reference to other links.
- **Links Referenced by** which is the container for all the links being referenced by other links.

Figure 3. WSA User Interface

The user interface of WSA is a web page, MainPage.jsp, composed with 3 frames. The top frame, webanalyzer.jsp, allows user to input the URL to be analyzed, the left frame will present the search result in a tree menu with folders and links inside. The right frame, FramesetRightFrame.jsp, will show the contents of the links in the left tree menu.

The design of the tree menu in left frame is explained. This is the core part of this tool.

The left frame is the main navigation index. There are three levels of folders that present the search results in a hierarchical tree menu.

The first level *Search Result* is the root folder. This folder itself is a hyperlink; it loads the search results summary page on the right frame by default, as well as allow user to click on it to switch back this page any time.

The second level of folders contain five folders in which each presents one category of links: Links Reference To, Links Reference By, Leaf Links, Dead Links, and Foreign Links.

The third level contains the links in each category. They are dynamically generated when user click on the second level of links. For example, link *X* is a link which can links to other links, then link X will be presented in the third level under folder *Links Reference To*. The link *X* itself is a folder that contains the actual files or links it links to. These files or links are presented at the last level and they are dynamic hyperlinks. For example, if *Y* can be referenced from *X*, then link *Y* will be presented under *X*. Click on Y will load the contents page of Y to the right frame or a new window.

5.3 Interaction between tiers

There are some interactions between different tiers, described below.

1. Client Server Interaction—Basic java networking HTML/HTTP
2. Server Running JSP and Sevlets—Java Applications
3. Server Database Interaction—JDBC-ODBC driver

Figure 4 and figure 5 illustrate the detailed relationship of files in different tiers that presents as JSP pages, Java class, or database. The detailed functionalities of these files are presented in the implementation section.

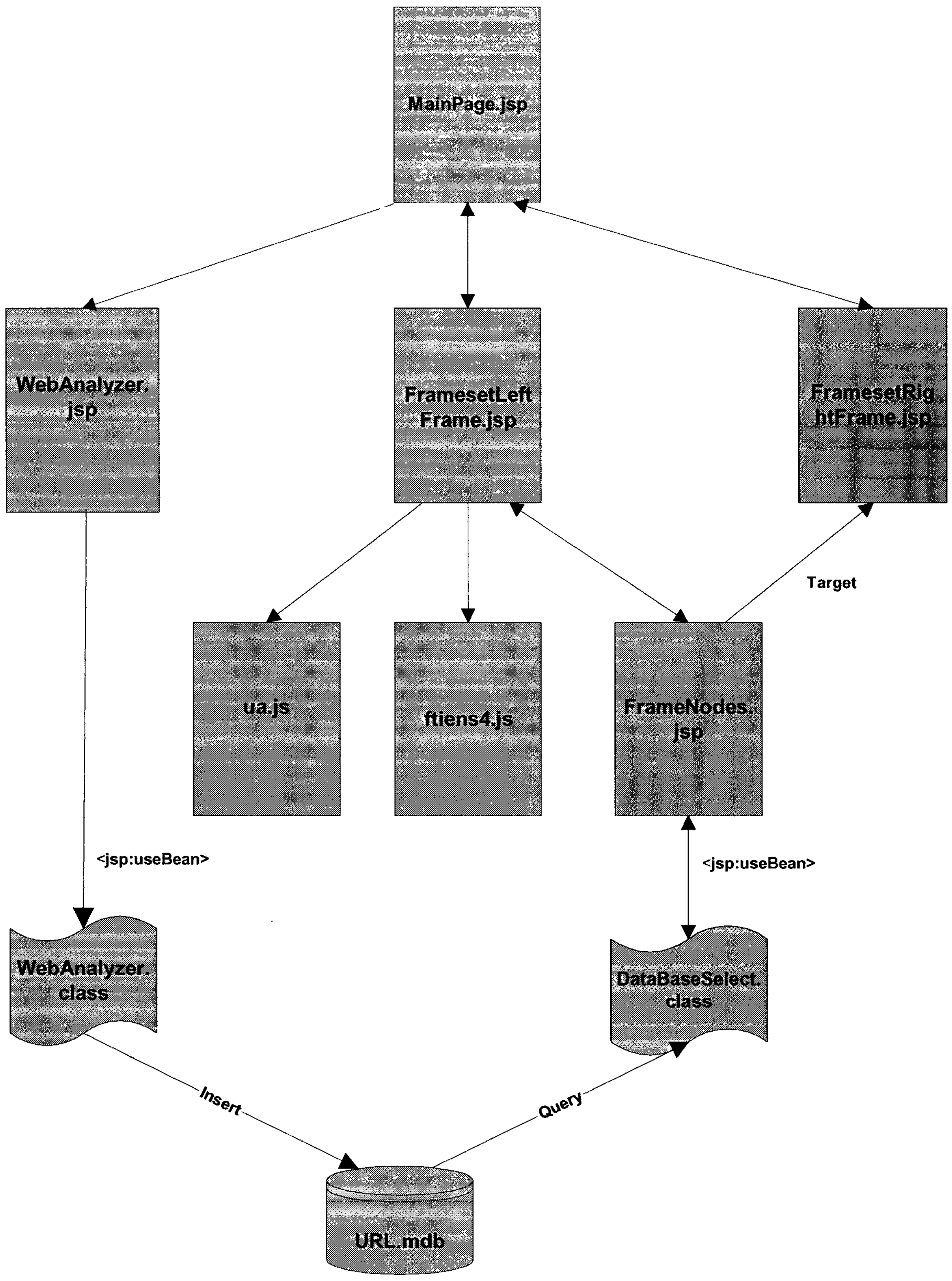


Figure 4. Files Relationship diagram

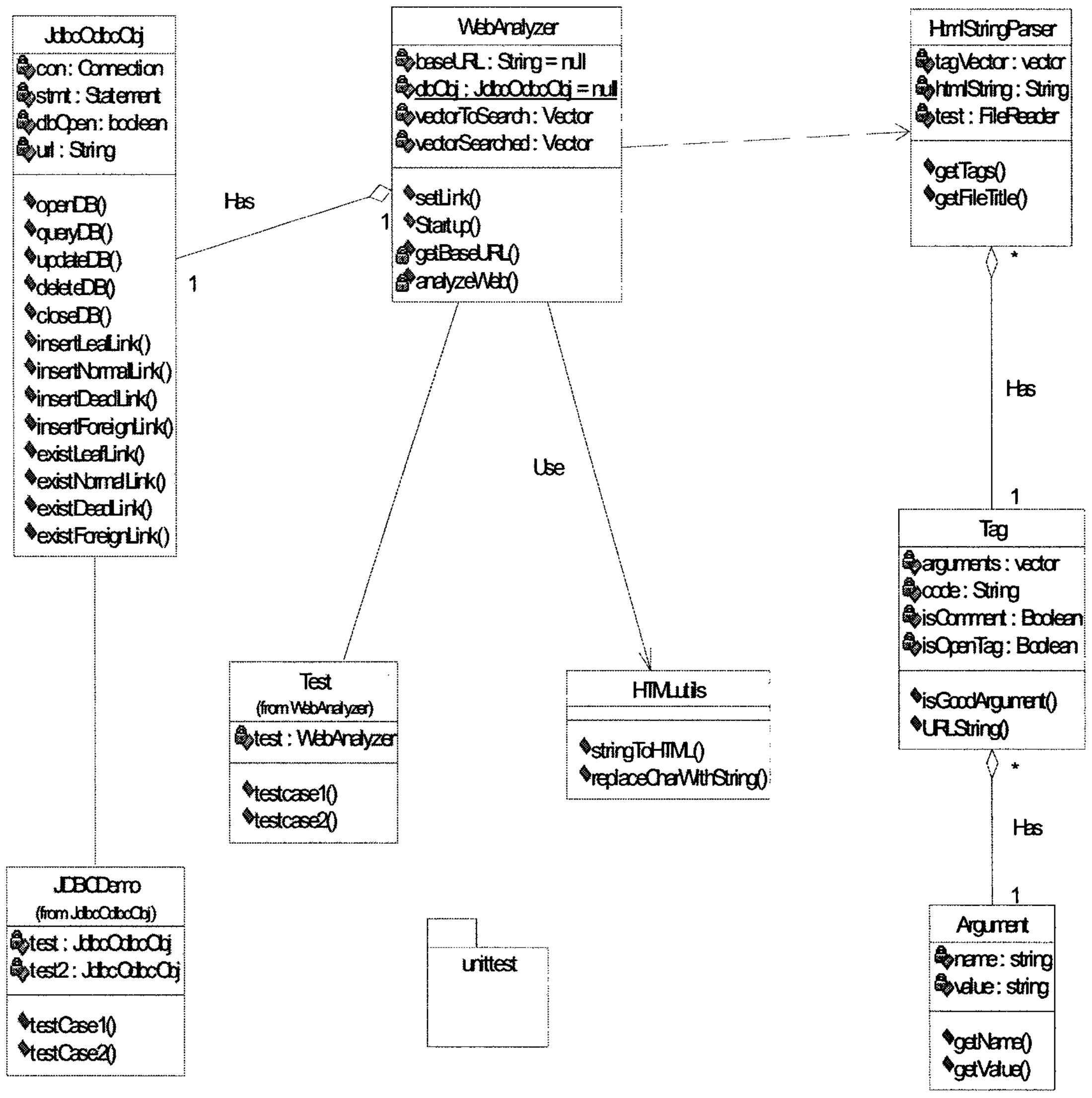


Figure 5 WebAnalyzer Class Diagram

6 Implementations

This section explains the implementations and functionalities of JSP pages and Java classes.

MainPage.jsp, is actually a HTML file with 3 framesets, the top frame allow user to input the URL name and submit the request. The left frame shows the detailed search result in a hierarchical tree menu, it is the core presentation area of WSA tool. The right frame shows the summary of the result by default, and it also shows the content page of the links in left frame.

Webanalyzer.jsp, the top frame, webanalyzer.jsp, accepts the user input, when user input a URL, http://www.cs.concordia.ca/programs/grad/diploma/courses.html, for example, and press submit button. It will go to call java WebAnalyzer.calss to process business logic, which include parse the link and search all the result, then store the results into the database. Here jsp:useBean tag was used, to call a bean which avoid using much java code in jsp pages.

```
<jsp:useBean id="webanalyzer" class="WebAnalyzer" scope="session"/>

<jsp:setProperty name="webanalyzer" property="*"/>
```

FramesetLeftFrame.jsp, the left frame, is the key part of the user interface. It performs three main tasks:

The first task is to load the browser detection code by calling ua.js; The second task is load to Treeview engine code ftiens4.js to construct the tree menu structure; The third task is to load the tree configuration file FramesetNodes.jsp to actually build the specific tree dynamically. Then the dynamic contents and the static contents are merged together, then return and show in the web browser. The detailed implementation is described below.

ua.js, which is to detect the browser, it supports all major browsers and versions, such as IE4.0+, NS4.7+, Mozilla 0.9+, Opera 7.0+.

ftiens4.js, which is the infrastructure code for the tree, and it is come with Treeview4.3. This file is written by JavaScript codes. It defines all the functionalities for tree configuration. For details please refer to http://www.treeview.net/ .

Function drawFolder(insertAtObj) implement the drawing of a folder, drawItem(insertAtObj) implement the drawing of the document, setStateFolder(isOpen) to open or closе a folder, addChild(childNode) to add new items under the a folder, gFld(description, hreference) is to create a new folder, gLnk(optionFlags, description, linkData) is to create a new link, insFld(parentFolder, childFolder) is to insert a folder under a folder, insDoc(parentFolder, document) is to insert a document or link to a folder.

FramesetNodes.jsp, called by FramesetLeftFrame.jsp, which is the code that actually builds the specific tree. It implements three tasks.

One is to call java bean and get the dynamic contents that will be used to build to specific tree; another is to specify layout options and other setting, then to actually build the tree with dynamic contents. This is the core file to present the search reports, so a detailed explanation of the implementation of this file will be presented. This will facilitate the understanding of how to use Treeview tool to build customised web pages with dynamic tree menu.

The first task, JSP page call java bean by following jsp tag:

```
<jsp:useBean id="queryDB" class="DataBaseSelect" scope="request">
</jsp:useBean>
```

By using useBean tag, it simplifies the JSP by reducing the amount of Java codes in JSP Pages. [14] Here, the id attribute specifies the name for a Bean—queryDB, it must be a unique throughout the page. The value of the class attribute specifies the class name of the JavaBean itself—DataBaseSelect in here. The scope attribute controls the a Bean's accessibility and life span it can have a value of page, request, session, or application. The accessibility of a Bean determines which pages or parts of a web application can access the Bean and its properties. A Bean's life span determines how long a particular Bean exists before it is no longer accessible to any page.

To interact with a Bean, the page is pointed to where to find the Java class file that defines the Bean and assigns it a name. This name will be used to access the values stored in the Bean's properties.

In this page, the queryDB.select is used to access the select function in DataBaseSelect class, and queryDB.getResult() to access getResult() function in DataBaseSelect. See this example:

```
<%queryDB.select("SELECT distinct Link from Link where Type = 'Normal'"); %>
<% Vector referenceTo = queryDB.getResult(); %>
```

this will load the category of referenceTo links results into the a vector *referenceTo* , then can be used to build the tree.

Before building the tree, the layout options are specified with other setting by assigning values to "environment variables". For example, STARTALLOPEN = 0 to set the tree is to start just showing the root folders, replace 0 with 1 to show the whole tree. ICONPATH = 'images/' where the gif was put in a subfolder, named images.

Finally, the tree was build with actual creation of folders and links. The process was as following:

1. To create the root folder:

   ```
   foldersTree = gFld("<i>Search Result</i>", "FramesetRightFrame.jsp")
   ```

   The function gFld takes a name "Search Result" with its format <i>--Italic font, an URL FramesetRightFrame.jsp, and returns the folder. The URL must either be just a file name (for example: FramesetRightFrame.jsp) or an whole URL complete with protocol , for example: http://www.cs.concordia.ca/logo.gif.

2. To place folders inside other folders by using the function 'insFld([parent folder], [child folder])'. For example:

   aux1 = insFld(foldersTree, gFld("Links Reference To", ""))

   There are five different types of links in WSA. They are Links Reference To, Links Reference By, Leaf links, Dead links, and Foreign links. So there are five first level folders which represent each category of links.

   The second level of folders will dynamically load the links of each category to under the folder it belongs to by using:

   aux2 = insFld(aux1, gFld("<% out.print(referenceTo.elementAt(i)); %>", "<% out.print(referenceTo.elementAt(i)); %>"))

3. To place the document inside the tree use the function 'insDoc([parentfolder], [document link])', for example:

   insDoc(aux2, gLnk("R", "<% out.print(target.elementAt(j)); %>", "<% out.print(target.elementAt(j)); %>"))

   which gLnk is to create a document link. gLnk takes three arguments: target, title, and Link. Detailed explain below:

   Target:

   'R' opens in the right frame,

   'B' opens in a new window,

   'T' opens in the current browser window, replacing the frameset if one exists,

   'S' opens in the current frame, replacing the tree.

Title: Text to be displayed in the tree for that node. This text can include some simple HTML, such as enclosing formatting tags (I, B, DIV, etc.))

Link: URL of the document (may be absolute or relative.) Do not enter other information in this string. Adding a target parameter or an event handler will not work.

The first argument "target" controls not only the target of the page (right frame, new window, etc.) but may also help in the construction of the link itself. After the character that controls the target (R,B,etc.), add an extra lowercase character to the string. This extra character will specify the protocol to be pre-pended to the URL. Example:

```
gLnk("R", "<% out.print(target.elementAt(j)); %>",
"<%out.print(target.elementAt(j)); %>"))
```

In the right frame there are two alternative pages: the default page is FramesetRightFrame.jsp, it contains two part of contents: Summary and Definitions. The summary presents the search results: the number of five categories links: Links Reference to, Links Reference By, Leaf Links, Dead Links, and Foreign Links.

This page will be changed when user click on the links in the left tree menu as mentioned above in gLnk([Target] , [Title], [Link]) function. Example:

```
 gLnk("R", "<% out.print(target.elementAt(j)); %>", "<%
out.print(target.elementAt(j)); %>"))
```

DatabaseSelect.class (DatabaseSelect.java), which implements:

To load ODBC-JDBC bridge driver, here is the main code:

```
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();
```

To obtain a database connection using DriverManager: [13]

```
        C = DriverManager.getConnection(url, username, password );
```

To connect and Query database, then to load results in a vector

```
         public String select(String queryString) {
           try {
                myResult = Stmt.executeQuery(queryString);
             result.clear();
                while (myResult.next()) {
                  result.addElement(myResult.getString(1));
                  }
             return "Connection Success!";
            } catch (SQLException E) {
                return "SQLException: " + E.getMessage();
                }
            }
```

Accessor for result. Accessor method used to obtain information about an object, the standard convention of accessor method names start with *get:*

```
                public Vector getResult() {
                 return result;
                }
```

Mutator for result, Mutator methods are essentially a request for an object to change its state, the standard convention for a mutator method names start with *set:*

```
          public void setResult(Vector avector) {
```

```
        result = avector;
        }
```

WebAnalyzer.class, this class implement parse the URL and input the search esults into the database. It includes a bund of inner classes.

URL.mdb is a Moorcroft Access database. In URL.mdb, there is only one table named Link, the attribute shows below:

Field Name	Data Type	Field Size	Description
Link	Text	100	Contains all the links search from the giving URL
Type	Text	50	Contains Normal, Dead, Leaf, Foreign types
Link_To	Text	100	Contains all the links referenced by links in field Link.

7 System Test and Results

7.1 System Requirement

Before trying to install and run WSA tool, check that the system meets the following requirements:

Hardware :

The program shall operate with the following hardware requirements:

- CPU 486 or later
- Monitor – SVGA (800x600) or latter
- RAM – 16 MB
- Disk Free Space 16M or more
- Mouse or equivalent pointing device

Software:

The following software should be set up in the machine

- JDK 1.3 or later
- Microsoft Access 98 or later
- Tomcat 3.2.4 or later

Platform:

The program can run on the following platforms

- Windows 98
- Windows NT
- Windows 2000

7.2 Analysis of a website

After successfully installed and start WSA, the main page of WSA appears. The user is required to input an URL (Uniform Resource Locator) into the text box in the top frame, and then press the Submit button to start searching.

The format of URL that WSA can analyze includes HTML, ASP, JSP, PHP pages and folders. Some example of URL could be:

http://www.cs.concordia.ca

http://www.cs.concordia.ca/programs/grad/diploma/courses.html

http://www.newatlanta.com/products/servletexec/index.jsp

http://www.cs.concordia.ca/~faculty/grogono/

https://www.mywebsite.org

www.mail.yahoo.com

When inputting the URL, the usual protocol http:// may be omitted. It is the default protocol of WSA. But the https:// cannot be omitted. After clinking on the submit button, it may take a few seconds or even several hours to analyze the website and save the results into the database. It depends on the size of the website being analyzed.

Once the progress bar the right bottom of explorer shows the search finished, the user can click on Refresh button on the top of explorer to update the search result in the tree menu. Clicking on the + or – can span or collapse the folder to navigate the tree.

One of the advantage of WSA is that it allows the user to partly analyze a website. The user can stop the analysis process any time they want by clicking on the icon Stop on the top of explorer. The results obtained so far will be stored in the database safely and displayed in the report tree correctly.

7.3 Test Cases and Experimental Results

In order to test the performance and capability of this tool, we chose various range of websites to test in term of size. From hundreds links size website to huge website engines. The experimentation proved that WSA could adapt small site as well as big size of web site with fast performance. One of the advantage of it is for very huge website, WSA allow user to partly analyze the website, that mean user can stop the searching process anytime by clicking stop button in explore. The results will exactly show the results of the finished part robustly.

1. http://www.cs.concordia.ca/programs/grad/diploma/courses.html

   Size: 476, Time: few seconds

2. http://www.cs.concordia.ca/~faculty/grogono/

   Size: 4082 Time: 2 minutes

3. http://www.yahoo.com

Since yahoo is a web sites engine, there it is impossible to finish in short periods, partly analyzed, the results was:

Got Size: 10000 Time: 7 minutes

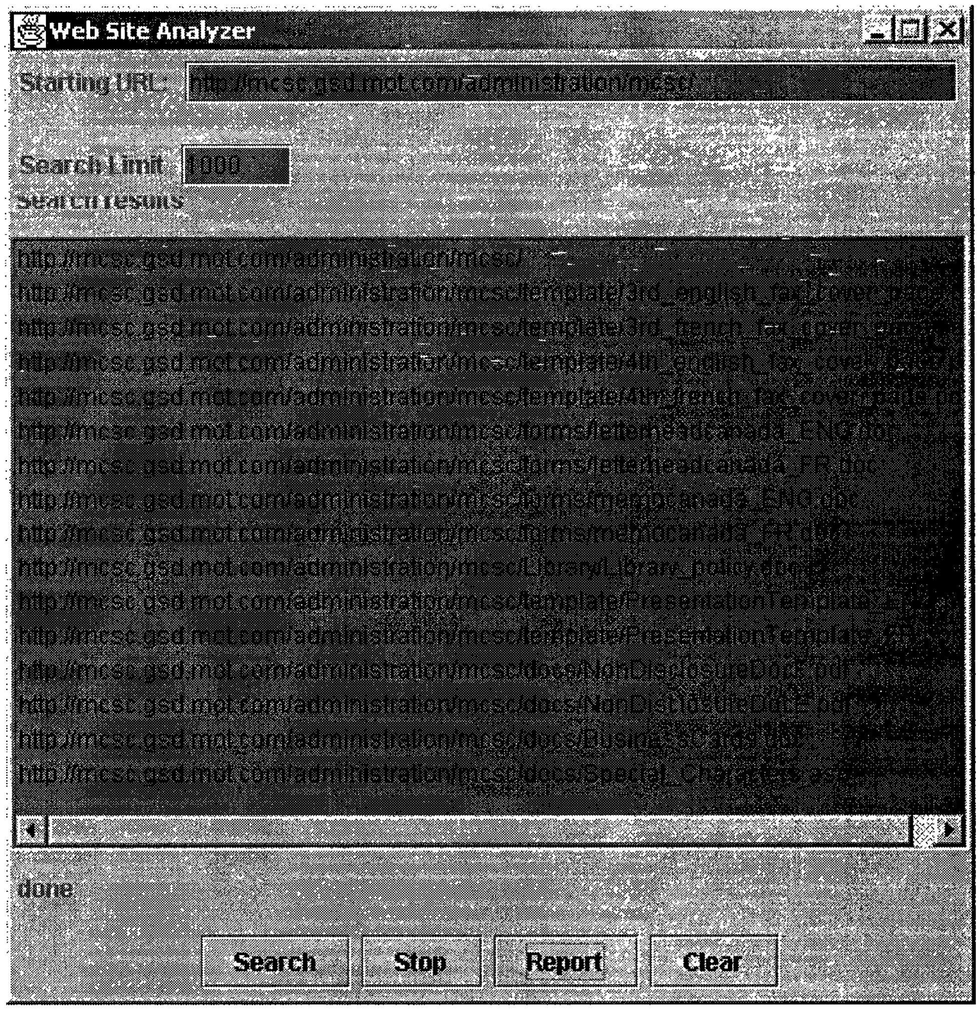
8 Related works

There are many kind of Web Site analyzer tools, such as Sitemapper, VIGOS,etc. Some of them analyze the contents of a web site by giving the detailed map with an indexed listing of all resources by page and category; some of them analyze the structure; some of them analyze the traffic of the web site for commercial propose.

8.1 Previous version of WSA

As mentioned in section 1.3 Project Scope, this project was based on the WSA C++ version implemented by Professor Peter Grogono and the Java version implemented by the Graduate student Yuan Xu [18] which called old WSA.

The old WSA version has two different graphic interfaces: One is the user input interface implemented in Java Swing, another one is the report interface in the static HTML file. They are showed as figure 6 and figure 7 below:



Starting URL: http://mcsc.gsd.mot.com/administration/mcsc/

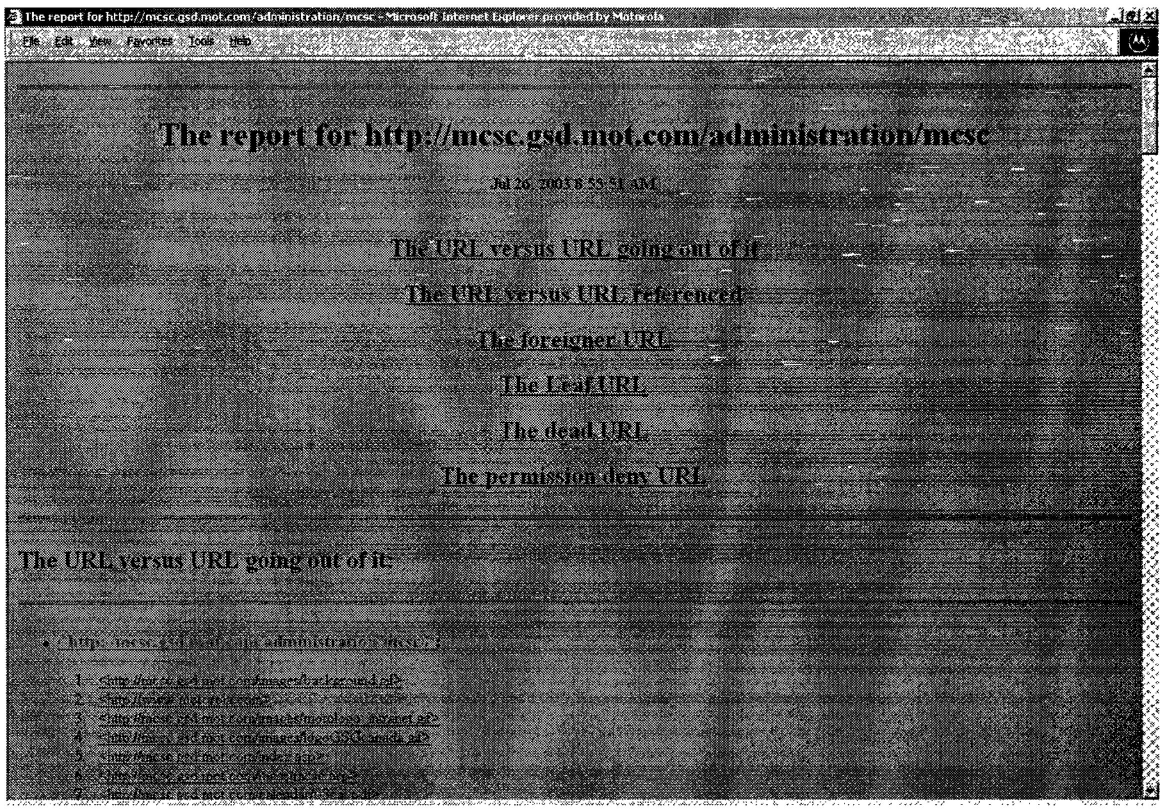
Search Limit: 1000

Search results

```
http://mcsc.gsd.mot.com/administration/mcsc/
http://mcsc.gsd.mot.com/administration/mcsc/template/3rd_english_fax_cover_pag
http://mcsc.gsd.mot.com/administration/mcsc/template/3rd_french_fax_cover_page
http://mcsc.gsd.mot.com/administration/mcsc/template/4th_english_fax_cover_page
http://mcsc.gsd.mot.com/administration/mcsc/template/4th_french_fax_cover_page.p
http://mcsc.gsd.mot.com/administration/mcsc/forms/letterheadcanada_ENG.doc
http://mcsc.gsd.mot.com/administration/mcsc/forms/letterheadcanada_FR.doc
http://mcsc.gsd.mot.com/administration/mcsc/forms/memocanada_ENG.doc
http://mcsc.gsd.mot.com/administration/mcsc/forms/memocanada_FR.doc
http://mcsc.gsd.mot.com/administration/mcsc/Library/Library_policy.doc
http://mcsc.gsd.mot.com/administration/mcsc/template/PresentationTemplate_E
http://mcsc.gsd.mot.com/administration/mcsc/template/PresentationTemplate_F
http://mcsc.gsd.mot.com/administration/mcsc/docs/NonDisclosureDocF.pdf
http://mcsc.gsd.mot.com/administration/mcsc/docs/NonDisclosureDocE.pdf
http://mcsc.gsd.mot.com/administration/mcsc/docs/BusinessCards.doc
http://mcsc.gsd.mot.com/administration/mcsc/docs/Special_Characters.as
```

done

Figure 6. User Interface of Previous WSA



The report for http://mcsc.gsd.mot.com/administration/mcsc - Microsoft Internet Explorer provided by Motorola

File Edit View Favorites Tools Help

The report for http://mcsc.gsd.mot.com/administration/mcsc

Jul 26, 2003 8:55:51 AM

The URL versus URL going out of it

The URL versus URL referenced

The foreigner URL

The Leaf URL

The dead URL

The permission deny URL

The URL versus URL going out of it:

- [illegible]

1. [illegible]
2. [illegible]
3. [illegible]
4. [illegible]
5. [illegible]
6. [illegible]
7. [illegible]

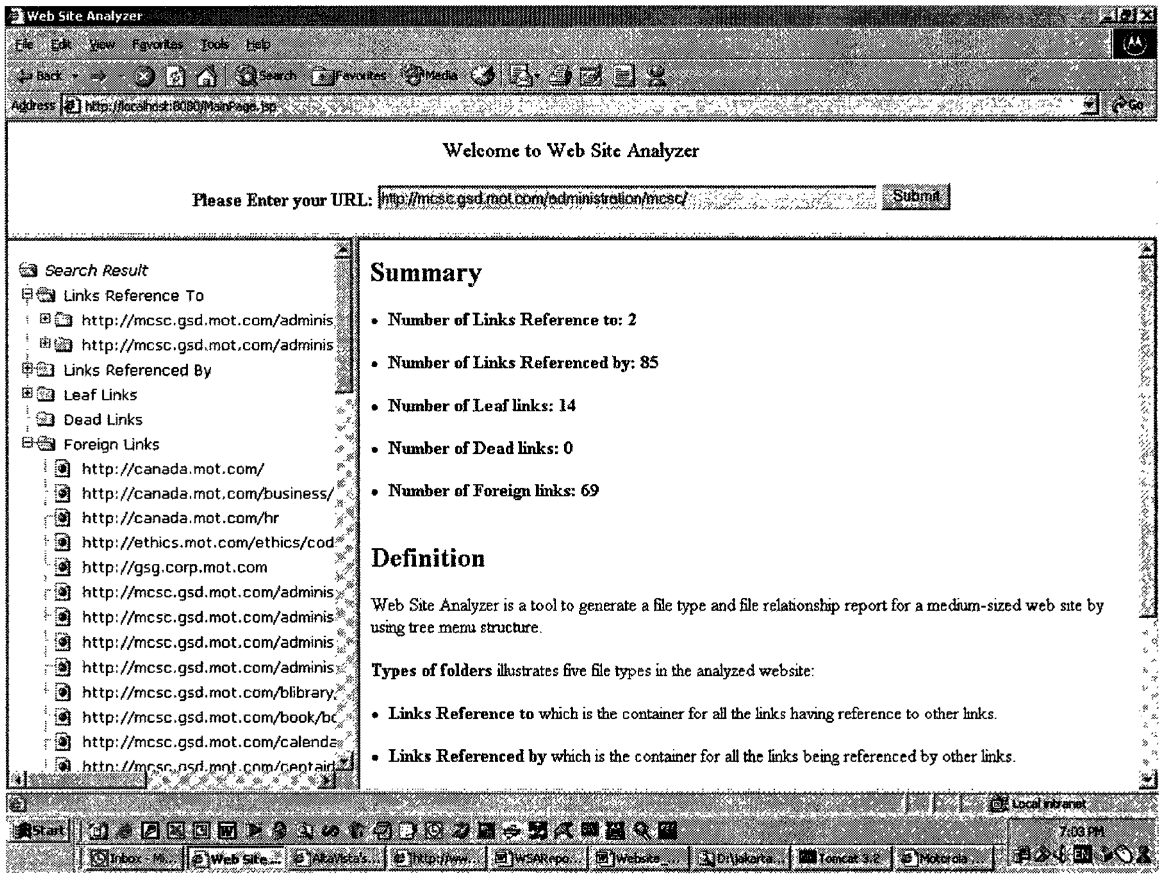
Figure 7. HTML reports of Previous WSA

This interface provides the user with a clear output and offers the user a friendly interface for input. However, the disadvantages of this GUI implementation are:

- The report was a plain HTML page without frames, so the relationship between links and link types are not straightforward.
- When clicking on each category to see the detailed information of this category or view the contents of a link, the high level view will be lost since the page is updated after the click action.
- The Report can be lengthy after analyzing the large web site and it will be hard to identify the place which the user want to go.

- The User input was from a Java Applet application and the report will be available in a HTML pages. This needs two GUI windows.
- The Report page is a static HMTL. The user cannot do any interaction in this page.

The new WSA solved all of these issues and it offers following advantages:


Web Site Analyzer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:8080/MainPage.jsp
Go
Submit
Local intranet
Start
7:03 PM

Welcome to Web Site Analyzer

Please Enter your URL: http://mcsc.gsd.mot.com/administration/mcsc/

Search Result
Links Reference To
http://mcsc.gsd.mot.com/adminis
http://mcsc.gsd.mot.com/adminis
Links Referenced By
Leaf Links
Dead Links
Foreign Links
http://canada.mot.com/
http://canada.mot.com/business/
http://canada.mot.com/hr
http://ethics.mot.com/ethics/cod
http://gsg.corp.mot.com
http://mcsc.gsd.mot.com/adminis
http://mcsc.gsd.mot.com/adminis
http://mcsc.gsd.mot.com/adminis
http://mcsc.gsd.mot.com/adminis
http://mcsc.gsd.mot.com/blibrary
http://mcsc.gsd.mot.com/book/bc
http://mcsc.gsd.mot.com/calenda
http://mcsc.gsd.mot.com/centair

Summary

- **Number of Links Reference to: 2**
- **Number of Links Referenced by: 85**
- **Number of Leaf links: 14**
- **Number of Dead links: 0**
- **Number of Foreign links: 69**

Definition

Web Site Analyzer is a tool to generate a file type and file relationship report for a medium-sized web site by using tree menu structure.

Types of folders illustrates five file types in the analyzed website:

- **Links Reference to** which is the container for all the links having reference to other links.
- **Links Referenced by** which is the container for all the links being referenced by other links.

Figure 8 new WSA user interface

1. "All in one" user interface. The user input and application outputs are all built in the same web page with three frames. This design is more coherent and user friendly. Figure 8 above shows new WSA user interface.

2. The use of frameset keeps the result links, the most important contents, are visible all the times. With menu controls as main navigation index, the results are presented in a hierarchical tree menu that includes folders and links underneath. This gives a very clear relationship between links of the given URL; meanwhile, user can easily span and close the folder to navigate the links.

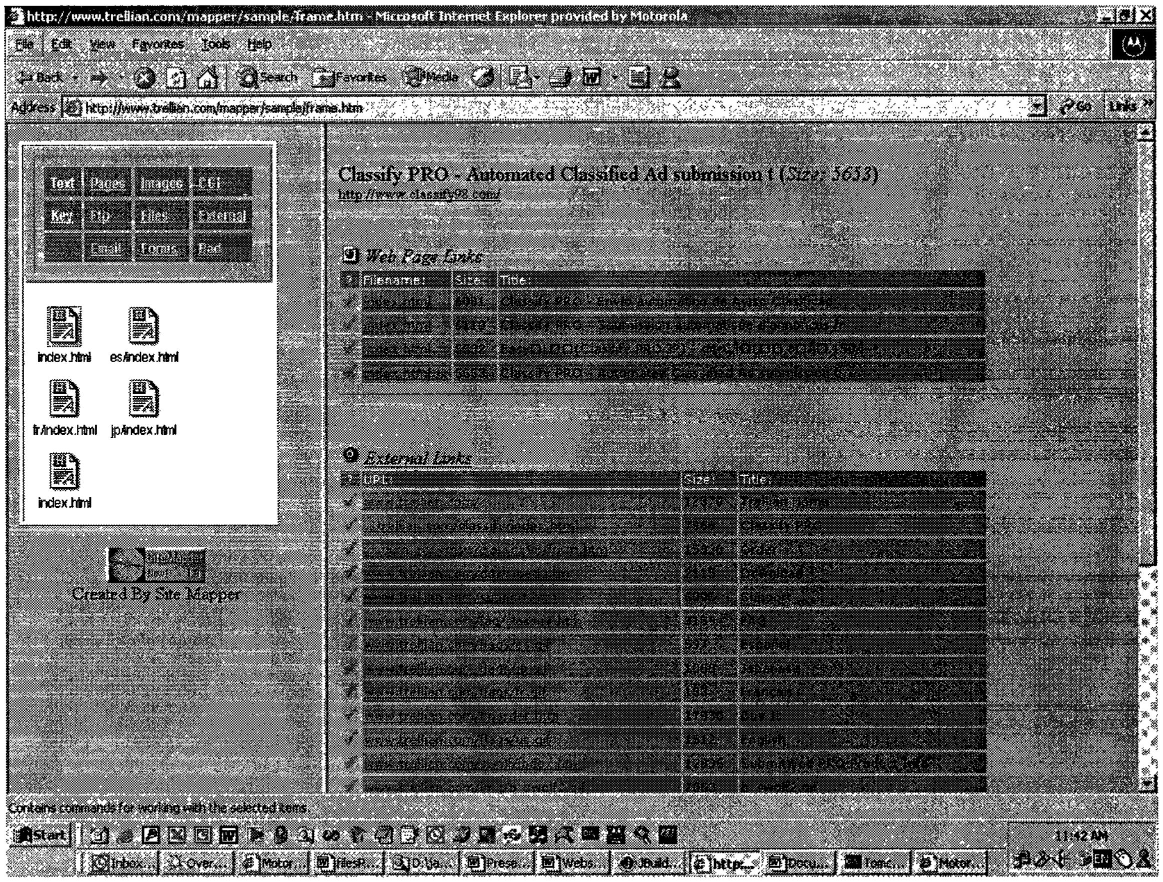
3. The database serialized the results; user can re-analyze the results anytime they want by just copying the database, without redo the search again. This also allows user to do the further analysis work off line.

4. Easy to change, easy to maintain. It offers the advantage of multi-tier application that is each tier can change at its own rate while minimizing the change to the entire application. [11]

8.2 Sitemapper

Site Mapper is a commercial Website Analyzer tool developed by company Trellian, it analyzes the contents of a website, and create a detailed map with an indexed listing of all resources by page and category. For more information, please refer to Site Mapper's home pages: http://www.submitwolf.net/sitemapper.html [16]

I installed and ran both Trellian Site Mapper version 1.04 and version 2.0 in my PC. The user interface in version 2.0 has been significant improved, which change the flat folders into hierarchy folders. This is the similar idea with WSA. Figure 9 and Figure 10 shows different user interfaces and search results of two versions of Site Mapper.


http://www.trellian.com/mapper/sample/frame.htm - Microsoft Internet Explorer provided by Motorola
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://www.trellian.com/mapper/sample/frame.htm
Go Links
Text Pages Images CGI
Key Ftp Files External
Email Forms Bad
index.html es/index.html
fr/index.html jp/index.html
index.html
Created By Site Mapper
Contains commands for working with the selected items.
Start
11:42 AM

Classify PRO - Automated Classified Ad submission t (*Size: 5653*)
http://www.classify98.com/

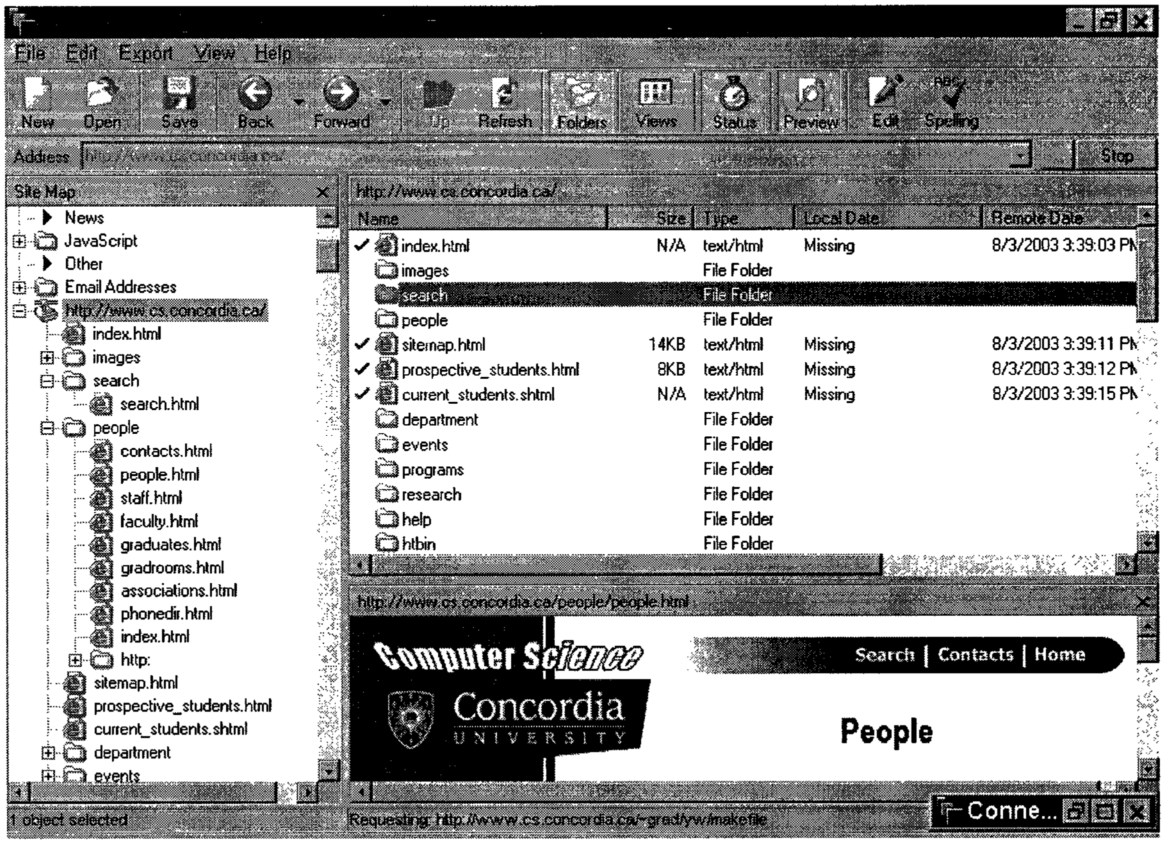
Web Page Links

?	Filename:	Size:	Title:
✓	index.html	6001	Classify PRO - [illegible]
✓	index.html	6113	Classify PRO - [illegible]
✓	index.html	5692	[illegible]
✓	index.html	5653	Classify PRO - Automated Classified Ad submission t

External Links

?	URL:	Size:	Title:
✓	www.trellian.com/	12379	Trellian Home
✓	[illegible]	7966	Classify PRO
✓	[illegible]	15320	Order
✓	[illegible]	2115	Download
✓	[illegible]	6090	Support
✓	www.trellian.com/faq/[illegible]	4155	FAQ
✓	www.trellian.com/[illegible]	997	Español
✓	www.trellian.com/[illegible]	1008	[illegible]
✓	www.trellian.com/[illegible]	1003	Français
✓	www.trellian.com/[illegible]	17930	Buy [illegible]
✓	www.trellian.com/[illegible]	1542	English
✓	www.trellian.com/[illegible]	12996	Submitwolf PRO [illegible]
✓	[illegible]	7053	[illegible]

Figure 9 Site Mapper 1.04 User Interface



Name	Size	Type	Local Date	Remote Date
✓ index.html	N/A	text/html	Missing	8/3/2003 3:39:03 PM
images		File Folder		
search		File Folder		
people		File Folder		
✓ sitemap.html	14KB	text/html	Missing	8/3/2003 3:39:11 PM
✓ prospective_students.html	8KB	text/html	Missing	8/3/2003 3:39:12 PM
✓ current_students.shtml	N/A	text/html	Missing	8/3/2003 3:39:15 PM
department		File Folder		
events		File Folder		
programs		File Folder		
research		File Folder		
help		File Folder		
htbin		File Folder		

Figure 10. Site Mapper 2.0 User Interface

Comparing the latest version of Site Mapper and WSA, both of them present similar idea of user interface, which is using the hierarchy directories menu control as their main navigation index, also the results are organized into different categories.

The difference is Site Mapper a commercial, multi propose Website Analyzer tool, provides the following features: Validates all links to prevent 'File not found' errors and broken images; Show JavaScript files, style sheets, media files etc; All results can be exported as text, XML or html documents; Spell check documents as they are mapped; Built in document preview functionality; Save files to local directory; View file properties [16]. Some of the features may not be useful for Web masters or personal web owner. Its multiple setting sometimes may be a cumbersome for users. This tool is something you have to learn how to use. On the other hand WSA is more simple and straightforward, user don't need to choose any options, the only operation for user is input the URL to be analyzed. There is no learning curve necessary.

Another advantage of WSA is it can be used in both single PC and local network while Site Mapper 2.0 exist some limitation. When there is a firewall in the network, it can still be used as long as the input link is inside the network. But as I used Site Mapper 2.0 in such a network, it failed to do any search and report empty link while it works fine in a personal PC that use a modem and dial to the Internet.

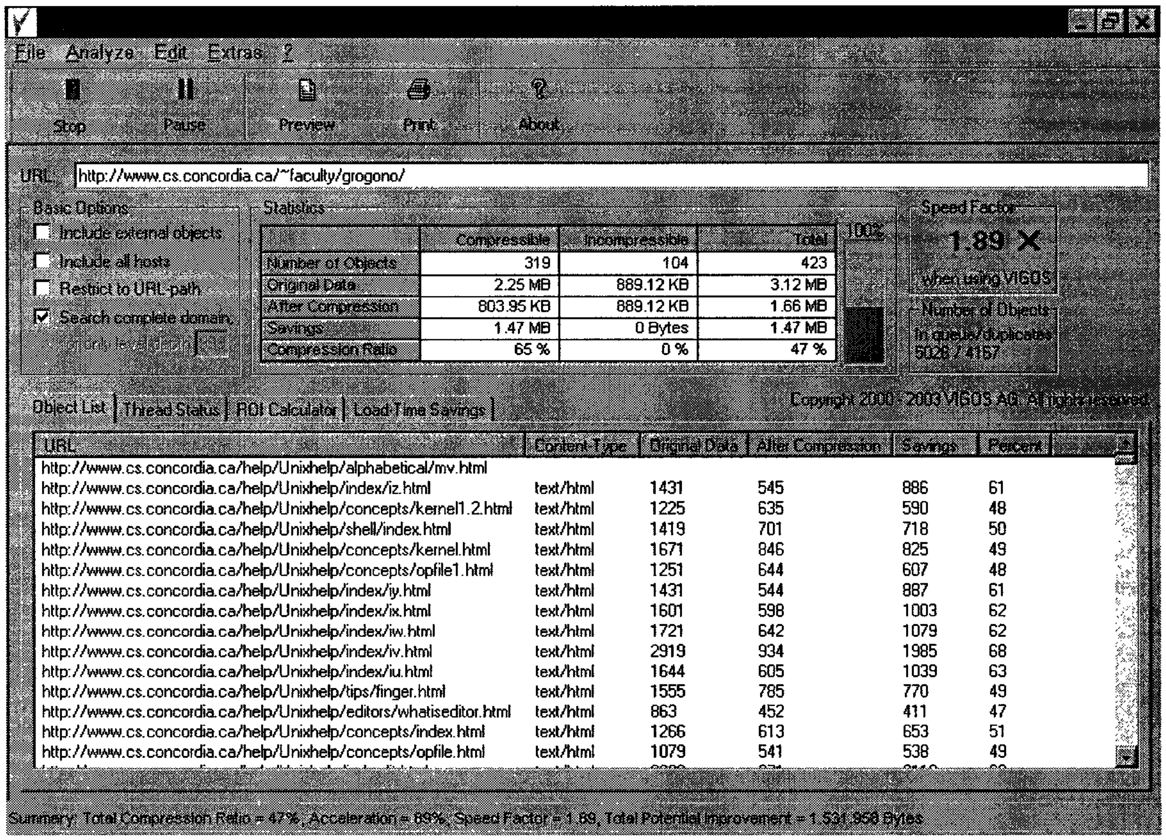
Additional, as mentioned above, Site Mapper is a commercial product. The price of SiteMapper2.0 electronic download version is $39.95 US.

8.3 VIGOS

The VIGOS Website Analyzer is a Free Website analyzing tool. When user type in an URL, the Website Analyzer will immediately start to spider the entire Website and estimate the potential Speed and Bandwidth improvements. [17] For more information, please refer to http://www.vigos.com/analyzer/ .

I installed and ran VIGOS in my PC. Figure 11 below shows the user interface and search results of analyzing http://www.cs.concordia.ca with this tool. It displays the VIGOS-

Compression of the various Website objects and generates Reports forecasting the transfer-cost reduction and page load-time acceleration. The reports listed in the flat HTML page. It does not show the hierarchy structure of the website and relationship between links as WSA does. Actually it is a tool more to monitor the Website Performance and Content Acceleration, less to analyze the Web Site structure.



URL: http://www.cs.concordia.ca/~faculty/grogono/

Basic Options:
- [ ] Include external objects
- [ ] Include all hosts
- [ ] Restrict to URL-path
- [x] Search complete domain

Statistics

	Compressible	Incompressible	Total
Number of Objects	319	104	423
Original Data	2.25 MB	889.12 KB	3.12 MB
After Compression	803.95 KB	889.12 KB	1.66 MB
Savings	1.47 MB	0 Bytes	1.47 MB
Compression Ratio	65 %	0 %	47 %

100%

Speed Factor: 1.89 X when using VIGOS

Number of Objects in queue/duplicates: 5028 / 4157

Object List | Thread Status | ROI Calculator | Load-Time Savings

Copyright 2000 - 2003 VIGOS AG. All rights reserved.

URL	Content-Type	Original Data	After Compression	Savings	Percent
http://www.cs.concordia.ca/help/Unixhelp/alphabetical/mv.html					
http://www.cs.concordia.ca/help/Unixhelp/index/iz.html	text/html	1431	545	886	61
http://www.cs.concordia.ca/help/Unixhelp/concepts/kernel1.2.html	text/html	1225	635	590	48
http://www.cs.concordia.ca/help/Unixhelp/shell/index.html	text/html	1419	701	718	50
http://www.cs.concordia.ca/help/Unixhelp/concepts/kernel.html	text/html	1671	846	825	49
http://www.cs.concordia.ca/help/Unixhelp/concepts/opfile1.html	text/html	1251	644	607	48
http://www.cs.concordia.ca/help/Unixhelp/index/iy.html	text/html	1431	544	887	61
http://www.cs.concordia.ca/help/Unixhelp/index/ix.html	text/html	1601	598	1003	62
http://www.cs.concordia.ca/help/Unixhelp/index/iw.html	text/html	1721	642	1079	62
http://www.cs.concordia.ca/help/Unixhelp/index/iv.html	text/html	2919	934	1985	68
http://www.cs.concordia.ca/help/Unixhelp/index/iu.html	text/html	1644	605	1039	63
http://www.cs.concordia.ca/help/Unixhelp/tips/finger.html	text/html	1555	785	770	49
http://www.cs.concordia.ca/help/Unixhelp/editors/whatiseditor.html	text/html	863	452	411	47
http://www.cs.concordia.ca/help/Unixhelp/concepts/index.html	text/html	1266	613	653	51
http://www.cs.concordia.ca/help/Unixhelp/concepts/opfile.html	text/html	1079	541	538	49

Summary: Total Compression Ratio = 47%, Acceleration = 89%, Speed Factor = 1.89, Total Potential Improvement = 1,531,958 Bytes

Figure 11. VIGOS User Interface

Comparing to VIGOS, WSA is better in analyzing the structure of the web site. It summaries the number of five categories of links as well as presents very clearly the relationships between links, while VIGOS fail to archive this goal.

The same as Site Mapper 2.0, VIGOS can be used in a personal PC that use a modem and dial to the Internet, while it may fail to do search and report empty link when used in a network.

9 Conclusion and Recommendation

9.1 Conclusion

WSA is a powerful Website analysis tool with a user-friendly interface and high performance. It can adapt to various ranges of size websites, and is especially suitable for small and middle size websites. It can be used in both single PC and local network.

Because of the loose coupling of the design, the user input and presentation output can work independently. That means one can input and perform search a URL once, and analyze the result many times as one wish without redo the search again, this also allows one to do the analysis work off line. The database serialized the results; user can re-analyze the results anytime they want by just copying the database.

Its "all in once" use interface maximum simplify user's operation, as well as presents the idea of "what you see is what you get"(WYSIWYG).

It demonstrates an example of JSP three-tier web applications design methodology. The main advantage is to hide the cooperative business logic from the user interface presentation.

9.2 Future works

There is a gap between user input and application output in this tool. After the user input the desire URL, the search results will be loaded in to the database. After it finished to load the database, the user needs to press the *Refresh* icon in the top of the IE to trigger the application to read the database and refresh the tree menu. A suggested solution is to implement a function behind the scene that audits the status of the database loading and automatically triggers the application to read the database and refresh the tree menu.

In this tool, the Orphan links have not been searched and reported yet, since the traverse algorithm cannot reach this type of links. The algorithm starts search from the given URL, then remember all the links and files it reach, then continue the search from these links and files, and so on. Orphan links are not connected to any these links and files, that why they cannot be reached. To solve this issue, a new algorithm is needed.

As mentioned in section 4.2, to simplify the interface, we designed an "All in One" interface that means that the user input and application output are all built in a frameset web page. The shortage of this design might be that for very large websites, it may take very long time to perform the search, though there is a progress bar showing the status in the right bottom which built in by IE browser. The user may not be informed how much time it will take and what many percentage has exactly been finished.

It would be nice to provide the ability to Pause and Resume the search. For huge web site, it may take couple hours to search the results, so the application has to keep on line. But this is not easy to implement since many data status must be remembered at the moment of Pause, it will be tricky for the dynamic web sites, because when the application re-connect to that web site, the relationship between some links or even the structure of that web site may have been changed.

10 Reference

[1] Grogono, Peter "Website Requirement"(ONLINE), Montreal, Quebec, CA: Concordia University, 2001, http://www.cs.concordia.ca/~faculty/grogono/webreqs.pdf, May 1, 2002.

[2] Hall, Marty "Core Servlets and JavaServer Pages", Upper Saddle River, NJ 07458, USA: Prentice Hall PTR, 2000. Pg. 10-11.

[3] Sun Microsystems Inc. "JavaServer Pages Dynamically Generated Web Content"(ONLINE), 1995-2003, http://java.sun.com/products/jsp/, June 7, 2003

[4] Fields, Duane K. and Kolb, Mark A. "Web Development with JavaServer Pages", Creenwich, CT06830, USA: Manning Publications Co. 2000.

[5] Hall, Marty "More Servlets and JavaServer Pages", Upper Saddle River, NJ 07458, USA: Prentice Hall PTR, 2001.

[6] Hanna, Phil "ASP2.0: The Complete Reference", NewYork: McGraw-Hill/Osborne, 2003. Pg. 22, 402

[7] "The Apache Jakarta Project http://jakarta.apache.org/"(ONLINE), Apache Software Foundation, 1999-2003, http://jakarta.apache.org/tomcat/, Feb 22, 2003

[8] Eaves, Jon Jones, Rupert and Godfrey, Warner "Apache Tomcat Bible" NewYork: Wiley Publishing Inc. 2003

[9] "Web Server Survey"(ONELINE), Nettraft ltd. 2003, http://news.netcraft.com/archives/web_server_survey.html, July 2, 2003

[10] Martins, Marcelino, "Treeview JavaScript: the only cross-browser DHTML tree with online visual builder"(ONLINE), http://www.treeview.net/, Feb 15, 2003

[11] "Enterprise Java 2, J2EE 1.3 Complete" SanFrancisco: SyBex Inc. 2003. ISBN: 0-7821-4145-5. Pg. 8

[12] Bergsten, Hans "JavaServer Pages Second Edition" U.S.A. : O'Reilly & Associates, Inc. ISBN 0-596-00317-X, 2001-2002. Pg. 125-127

[13] Spell, Brett "Professional Java Programming", Birmingham,UK: Wrox Press Ltd, 2000. ISBN 186100382X

[14] Darwin, Ian F. "Java Cook book" U.S.A: O'Reilly, 2001. ISBN: 0-596-00170-3.

[15] "JDBC Data Access API"(ONLINE), Sun Microsystems, Inc. 1995-2003. http://java.sun.com/products/jdbc/ June 14, 2003

[16] "Sitemapper"(ONLINE) Trellian Inc. 1997 – 2003, http://www.trellian.com/mapper/index.html, July 25, 2003

[17] "Free Website Compression Benchmarking Tool Website Analyzer"(ONLINE) http://www.vigos.com/analyzer/, July 25, 2003

[18] Xu, Yuan , "WEB SITE ANALYZER" Montreal, Quebec, CA: Concordia University, 2002.

Appendix WSA Installation Guideline

Step1: Installing Jakarta-Tomcat, if it is already installed, omit this step.

- Download Jakarta-Tomcat from:

  http://jakarta.apache.org/builds/jakarta-tomcat/release/v3.2.4/bin/jakarta-tomcat-3.2.4.zip.

- Unzipped it to Drive D:\ and new folder D:\jakarta-tomcat-3.2.4 will appear.

Step2: Copy WSA files to the PC

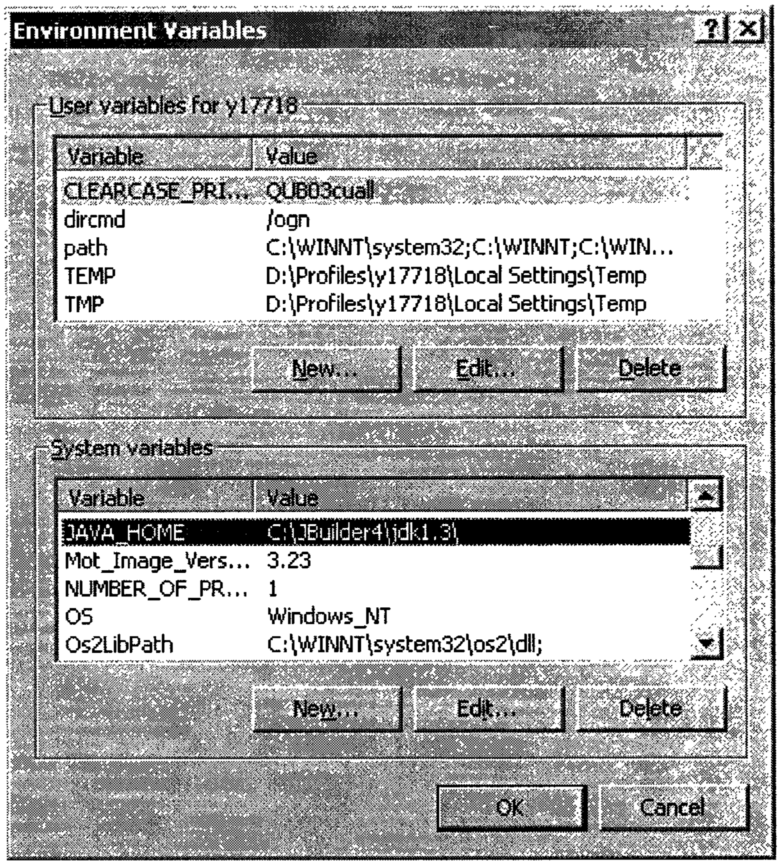
File location should be:

Directory	***Files***
D:\jakarta-tomcat-3.2\webapps\ROOT	JSP, HTML, DB file
D:\jakarta-tomcat-3.2\webapps\ROOT\WEB-INF\classes	classes files
D:\jakarta-tomcat-3.2\webapps\ROOT\images	image files

Step3: Set windows system environment variables

- Run start->settings->control panel, control panel window appears.
- Click 'system', and 'System Properties' window appears. Go to "advance" tab
- Click 'Environment variable' button, go to "System Variable". (or User Variable if no self administration rights for that PC or workstation, in this case, reset the environment variables at re-login)

- Set the JDK home: click New button, type 'JAVA_HOME' in 'Variable' field, and 'C:\JBuilder4\jdk1.3\' (or JDK directory in the PC) in 'Value' field, then click 'OK' button to add it. It will be shown as below:



User variables for y17718

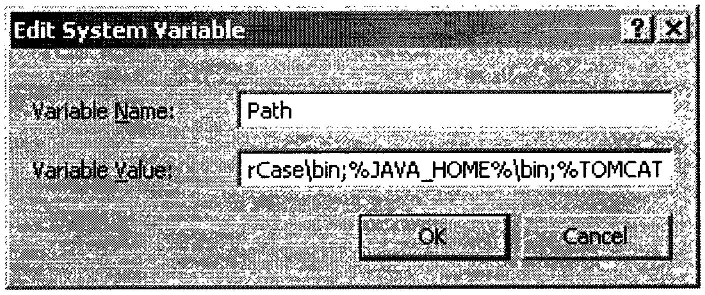
Variable	Value
CLEARCASE_PRI...	QUB03cuall
dircmd	/ogn
path	C:\WINNT\system32;C:\WINNT;C:\WIN...
TEMP	D:\Profiles\y17718\Local Settings\Temp
TMP	D:\Profiles\y17718\Local Settings\Temp

System variables

Variable	Value
JAVA_HOME	C:\JBuilder4\jdk1.3\
Mot_Image_Vers...	3.23
NUMBER_OF_PR...	1
OS	Windows_NT
Os2LibPath	C:\WINNT\system32\os2\dll;

- Set Tomcat home: click New button, type 'TOMCAT_HOME' in 'Variable' field, type "D:\jakarta-tomcat-3.2.4" (Tomcat directory) in 'Value' field and click 'OK' button to add it.

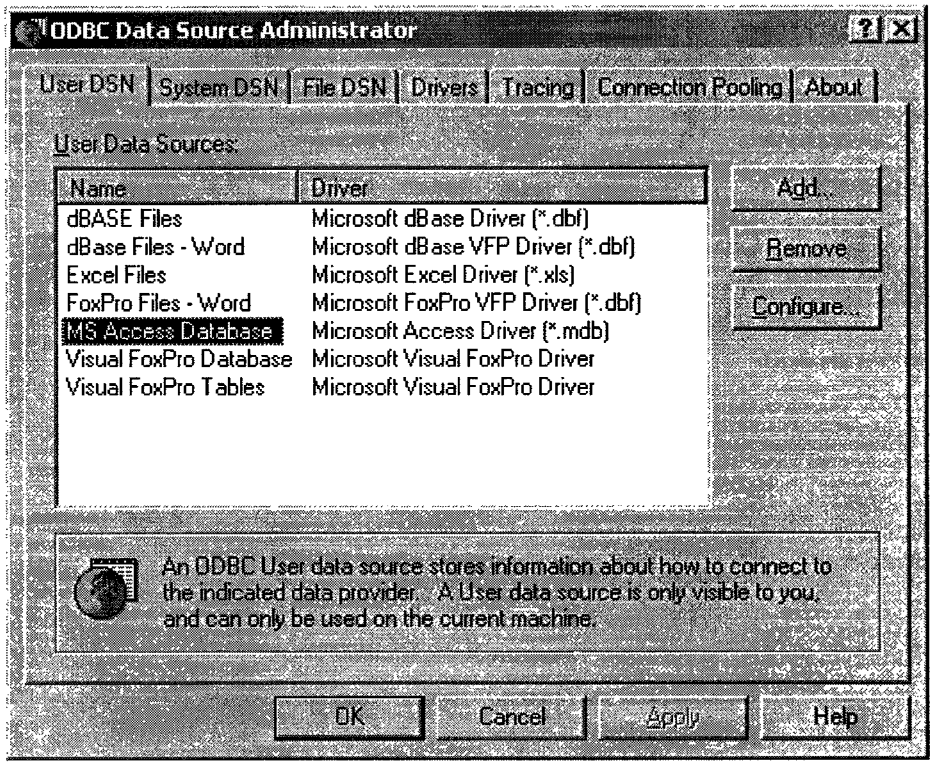
- Set PATH: click 'Path' on the 'System Variables" zone (or User Variable if no self administration rights), append ';%JAVA_HOME%\bin;%TOMCAT_HOME%\classes ' to the Value string, and click 'OK' button.



Variable Name: Path
Variable Value: rCase\bin;%JAVA_HOME%\bin;%TOMCAT

Step4: DSN Setup (JDBC- ODBC)

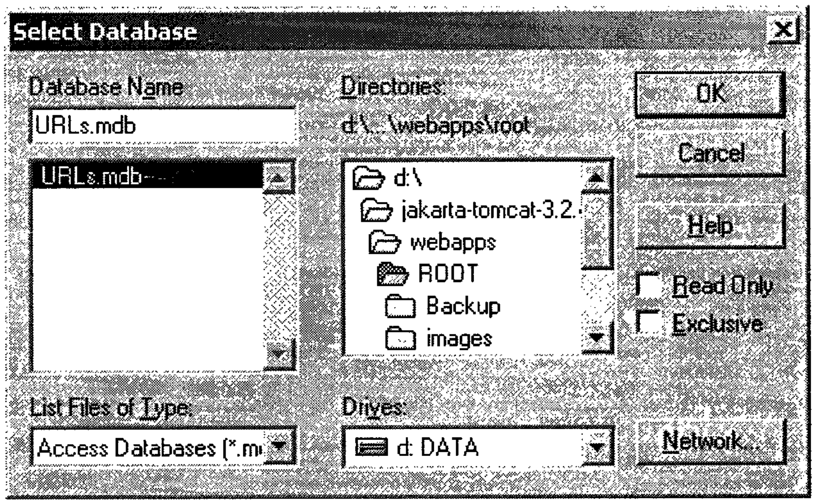
- Start→ setting→ control panel→ administrative tools→ data source (ODBC), choose MS Access Dababase and click on Add button as shown below.



Name	Driver
dBASE Files	Microsoft dBase Driver (*.dbf)
dBase Files - Word	Microsoft dBase VFP Driver (*.dbf)
Excel Files	Microsoft Excel Driver (*.xls)
FoxPro Files - Word	Microsoft FoxPro VFP Driver (*.dbf)
MS Access Database	Microsoft Access Driver (*.mdb)
Visual FoxPro Database	Microsoft Visual FoxPro Driver
Visual FoxPro Tables	Microsoft Visual FoxPro Driver

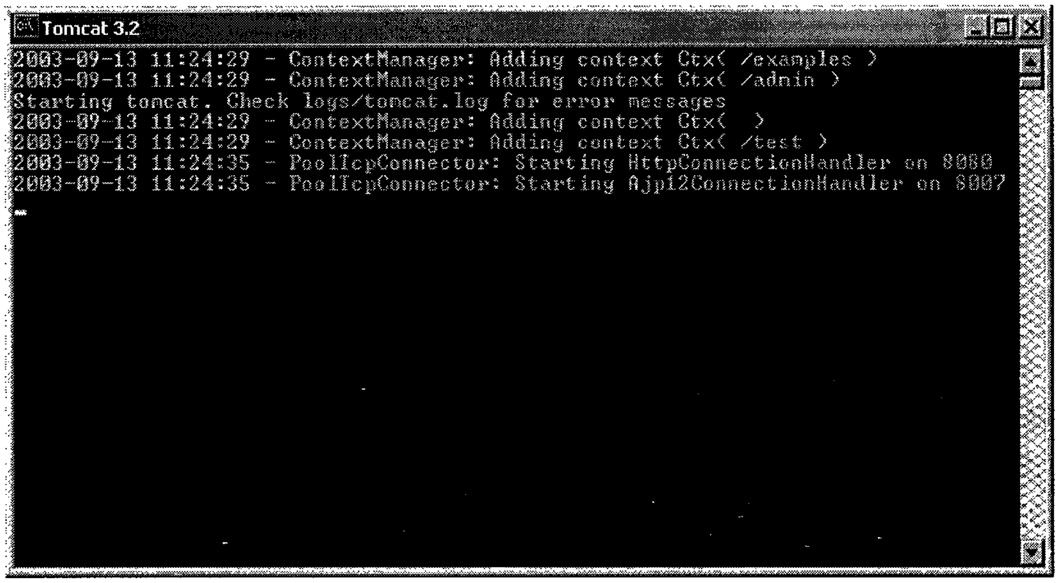
An ODBC User data source stores information about how to connect to the indicated data provider. A User data source is only visible to you, and can only be used on the current machine.

- Input Data Source Name: URLs, and click on select button, then select a database from the dialog box as shown below:



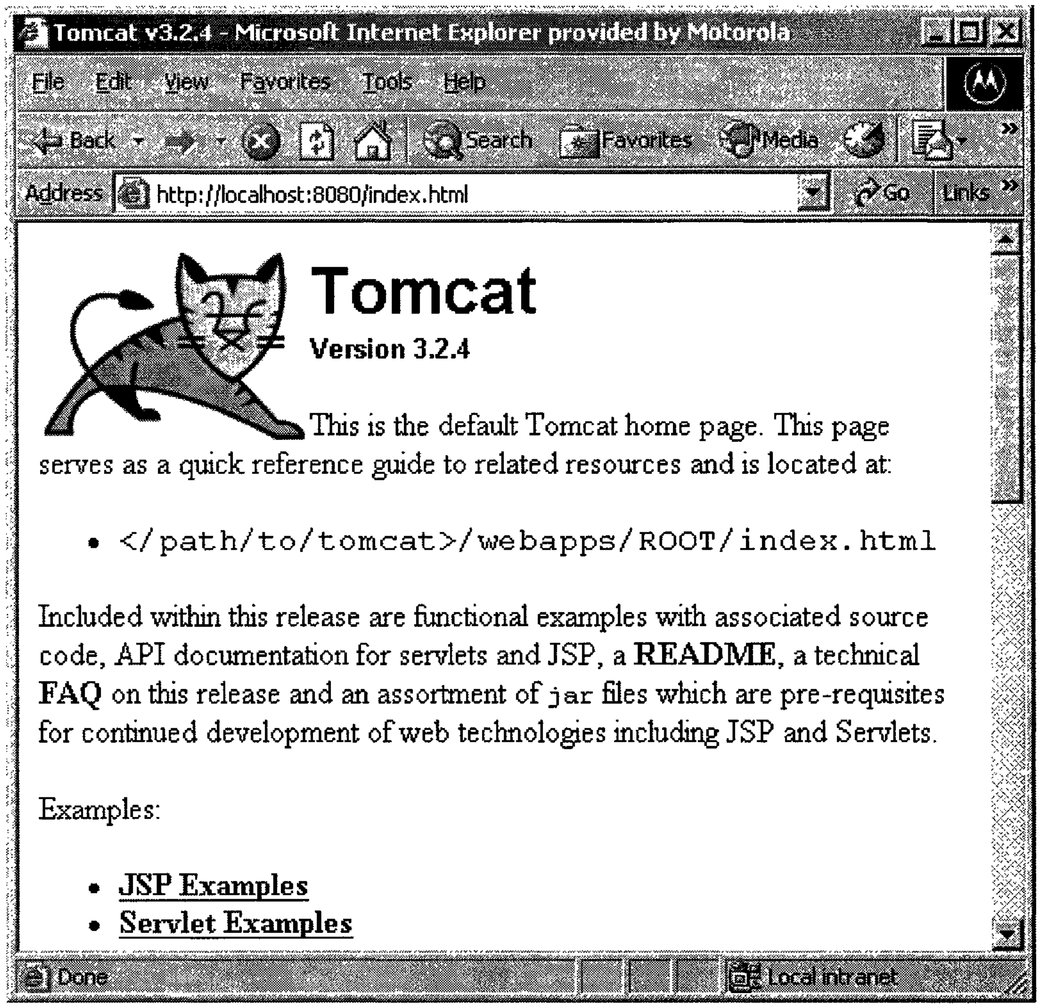
Step5: Start Tomcat Server

- Go to D:\jakarta-tomcat-3.2.4\bin folder and execute startup.bat. The window below should be shown up.



```
2003-09-13 11:24:29 - ContextManager: Adding context Ctx( /examples )
2003-09-13 11:24:29 - ContextManager: Adding context Ctx( /admin )
Starting tomcat. Check logs/tomcat.log for error messages
2003-09-13 11:24:29 - ContextManager: Adding context Ctx(  )
2003-09-13 11:24:29 - ContextManager: Adding context Ctx( /test )
2003-09-13 11:24:35 - PoolTcpConnector: Starting HttpConnectionHandler on 8080
2003-09-13 11:24:35 - PoolTcpConnector: Starting Ajp12ConnectionHandler on 8007
```

- Launch Internet Explorer Browser, type http://localhost:8080/ (Tomcat has its own HTTP server which listens on port 8080: default value), the page below should be shown up.


Tomcat v3.2.4 - Microsoft Internet Explorer provided by Motorola
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:8080/index.html
Go Links
Done
Local intranet

Tomcat

Version 3.2.4

This is the default Tomcat home page. This page serves as a quick reference guide to related resources and is located at:

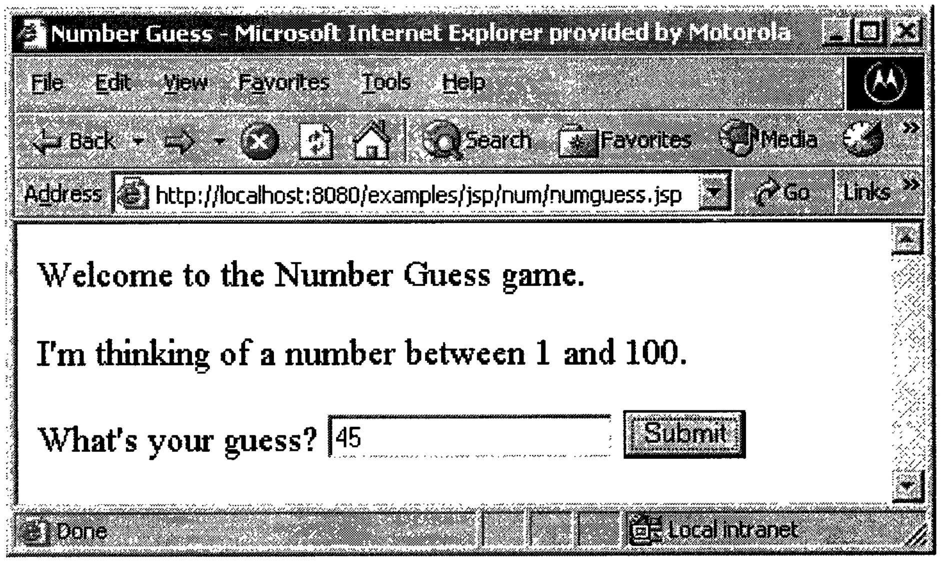
- `</path/to/tomcat>/webapps/ROOT/index.html`

Included within this release are functional examples with associated source code, API documentation for servlets and JSP, a **README**, a technical **FAQ** on this release and an assortment of `jar` files which are pre-requisites for continued development of web technologies including JSP and Servlets.

Examples:

- **JSP Examples**
- **Servlet Examples**

- Click on JSP Examples links and choose a example Numberguess to test Tomcat server. If the Tomcat server works fine, the below page should be shown up.



Number Guess - Microsoft Internet Explorer provided by Motorola
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:8080/examples/jsp/num/numguess.jsp Go Links
Submit
Done
Local intranet

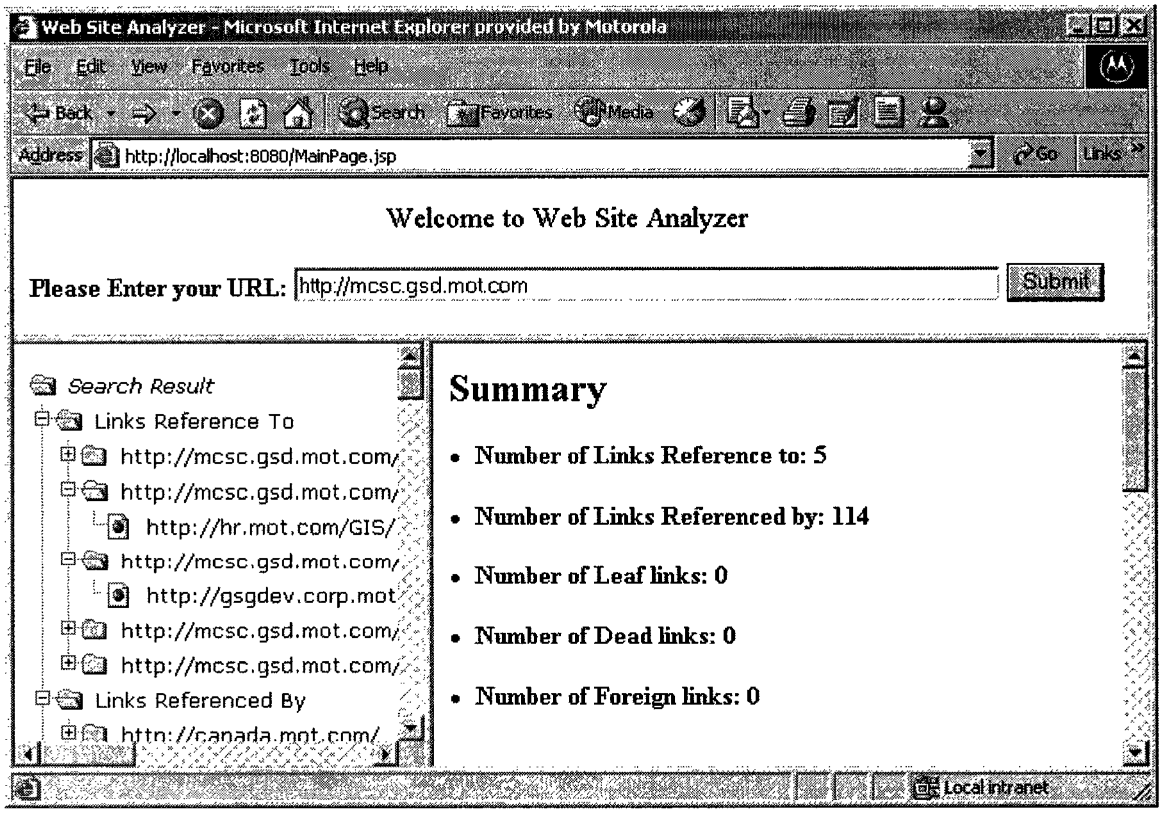
Welcome to the Number Guess game.

I'm thinking of a number between 1 and 100.

What's your guess? 45

Step6: Run WSA

- Type http://localhost:8080/MainPage.jsp , the WSA user interface will be shown up.



Web Site Analyzer - Microsoft Internet Explorer provided by Motorola
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:8080/MainPage.jsp Go Links
Submit
Local intranet

Welcome to Web Site Analyzer

Please Enter your URL: http://mcsc.gsd.mot.com

```
Search Result
  Links Reference To
    http://mcsc.gsd.mot.com/
    http://mcsc.gsd.mot.com/
      http://hr.mot.com/GIS/
    http://mcsc.gsd.mot.com/
      http://gsgdev.corp.mot
    http://mcsc.gsd.mot.com/
    http://mcsc.gsd.mot.com/
  Links Referenced By
    http://canada.mot.com/
```

Summary

- **Number of Links Reference to: 5**
- **Number of Links Referenced by: 114**
- **Number of Leaf links: 0**
- **Number of Dead links: 0**
- **Number of Foreign links: 0**

- Input the URL to be analyzed and click Submit button, the application will start to search. After it finishes searching, click on Refresh button on the top of the browser to update the results.