ABSTRACT

Investigation of Automated Population and Maintenance of a Resource Database using Web-based Screen Scraping and Web Services

By Subalekha Udayasankar

The World Wide Web (WWW) is a vast collection of files accessible to the public primarily through hyper-linked files called Web pages. Increasingly, people turn to the Web to locate suppliers, products and services.

The Community Disaster Information System (CDIS), developed by Miami University for the American Red Cross, provides Red Cross workers with a database of suppliers of products and services for disaster response. Currently, Red Cross Workers populate and maintain the CDIS database through manual data entry. The Web can be used for automating the data population and maintenance tasks in CDIS.

The goal of the thesis is to develop automated techniques for entering and maintaining the supplier information to enhance the quantity and quality of information in the CDIS system. Web-based screen scraping and Web Services are used to confirm that a significant portion of the data in the CDIS system can be automatically populated and maintained using the Web. The results suggest that using this approach, CDIS can store current, standard supplier information free of duplicates. The major drawback is that Web-based screen scraping requires maintenance in long-term usage.
Investigation of Automated Population and Maintenance of a Resource Database using Web-based Screen Scraping and Web Services

A Thesis

Submitted to the Faculty of Miami University
in partial fulfillment of
the requirements for the degree of
Master of Science
Department of Computer Science and Systems Analysis
by
Subalekha Udayasankar
Miami University
Oxford, Ohio
2007

Advisor ____________________________________________
Douglas Troy

Reader ____________________________________________
Scott Campbell

Reader ____________________________________________
Mike Helmick
## Table of Contents

1. Introduction..................................................................................................................... 1
2. Background ..................................................................................................................... 3
   2.1 Web-based screen scraping ....................................................................................... 3
   2.2 Web Services ............................................................................................................ 3
   2.3 CDIS Project ............................................................................................................. 6
   2.4 Related Research....................................................................................................... 8
3. Research Methodology ................................................................................................... 9
   3.1 Experiments for Automated Population ................................................................. 10
      3.1.1 Web-based Screen Scraping Methodology ...................................................... 11
      3.1.2 Web Services Methodology ............................................................................. 12
   3.2 Experiments for Automated Maintenance ................................................................ 13
   3.3 Standardization Technique ..................................................................................... 14
   3.4 Duplicate Detection Technique .............................................................................. 15
   3.5 Measures of Analysis .............................................................................................. 15
4. Implementation ............................................................................................................. 17
   4.1 Implemented Classes .............................................................................................. 17
   4.2 Automated Population ............................................................................................ 20
      4.2.1 Web-based Screen Scraping ............................................................................ 23
      4.2.2 Web Services ................................................................................................... 26
   4.3 Automated Maintenance ......................................................................................... 27
5. Results ........................................................................................................................... 30
   5.1 Automated Population ............................................................................................ 30
   5.2 Automated Maintenance ......................................................................................... 31
   5.3 Discussion ............................................................................................................... 31
6. Future Work .................................................................................................................. 33
7. Conclusion ..................................................................................................................... 34
Appendices ........................................................................................................................ 35
   Appendix A – Apache Axis .......................................................................................... 36
   Appendix B - USPS Standard Abbreviations ............................................................... 40
   Appendix C - Category Mapping .................................................................................. 41
Bibliography ..................................................................................................................... 43
List of Figures

Figure 1: Web Services Architecture ................................................................. 5
Figure 2: CDIS Record ....................................................................................... 7
Figure 3: Two Sets of Experiments ................................................................. 9
Figure 4: Steps Involved in Web-based Screen Scraping .............................. 12
Figure 5: Steps involved in population using Web Services ..................... 13
Figure 6: Automated Maintenance ............................................................... 14
Figure 7: Class Diagram I ............................................................................. 18
Figure 8: Class Diagram II .............................................................. 19
Figure 9: Class Diagram III ................................................................. 20
Figure 10: Flowchart of Automated Population .................................. 21
Figure 11: Flowchart of Automated Maintenance .................................. 28

List of Tables

Table 1: Strategy to classify retrieved suppliers ......................................... 22
Table 2: Strategy to maintain suppliers ......................................................... 29
Table 3: Automated Population analysis .................................................... 30
Table 4: Automated Maintenance Results .............................................. 31
1. Introduction

The American Red Cross is a federally mandated agency that responds to local and national disasters within the United States. Red Cross workers respond to disasters by providing aid through supplies and services to the victims. Examples of such aid include temporary shelter, food, clothing, medication and transportation, relocation support, finding animal shelter and locating replacement medical suppliers. The Community Disaster Information System (CDIS) [1] is an Internet-based database used by Red Cross chapters for storing information on the suppliers of goods and services for disaster response. Currently Red Cross volunteers must manually enter and maintain the information in the CDIS system. Some of the challenges faced by the Red Cross workers in maintaining the data in CDIS are to be able to retrieve information about new resources from the Web and to be able to successfully maintain the accuracy of the information that is already available in the CDIS database.

The World Wide Web Consortium (W3C) says, “The World Wide Web is the universe of network-accessible information, the embodiment of human knowledge” [31]. Thus, the Web should provide the potential for locating and maintaining information about the CDIS suppliers. This information can be found at directory service Web sites, such as Yellowpages.com, and at Web services, such as Yahoo local.

Web-based screen scraping and Web Services are some of the approaches that can be used to automatically retrieve this information from the web. Web-based screen scraping refers to the technique of parsing the markup language and extracting information from a Web page. Much of the information on the WWW is written using the HyperText Markup Language (HTML). HTML is a language that is used for creating Web pages. The markup tags give the browser information on how the information is to be displayed. Alternatively, Web Services provide a mechanism for allowing software applications to interoperate with one another over a network.

This is an investigation to develop automated techniques for entering and maintaining the supplier information to enhance the quantity and quality of information in the CDIS system. In order to examine the possibility of automated population, a series of experiments are performed. The first set of experiments involves Web-based screen
scraping and Web Services to extract the necessary information about the suppliers from the Internet and to organize them into various categories in a database. The second set of experiments involves comparing the database obtained through the first set of experiments with the current CDIS database and be able to add new information and maintain the existing information.

The success of these experiments will provide the CDIS system architecture for automated data population and maintenance. When used in CDIS, it will reduce manual date entry and maintenance to a vast extent and the accuracy of the information will make the system more useful and reliable.
2. Background

This section gives a brief overview of the techniques used in this research.

2.1 Web-based screen scraping

The ad-hoc technique, widely used in 80’s of extracting information from the display output is known as ‘screen scraping’. “The screen-scraping applications merely placed more attractive front ends on ugly, hard-to-learn and difficult-to-use mainframe systems” [3]. The advent of WWW allowed screen scraping to evolve into Web-based screen scraping. Web-based screen scraping refers to the technique of parsing the HTML and extracting information from the Web pages.

The creation of screen scraping software for a Web page requires the programmer to examine the HTML source of the Web page and identify the information and patterns required for extraction. The next step is to develop the code to extract and store the data. Existing HTML parsers can be used to make the task of screen scraping simpler. For example, if the Web page is in HTML, using a HTML parser makes extraction of information just a matter of traversing through the nested tags and locating the correct tag that contains the data after which the data can be parsed and extracted. Web-based screen scraping faces two major drawbacks. The first drawback is that it needs a human to read the source of the Web page and understand it before creating the scrapper. The second drawback is that any structural changes to the Web page will cause the scrapper to fail [2]. It then requires human intervention to redesign the scrapper according to the new structural changes and this affects the automation of the process. Web-based screen scraping is practical when used on a short-term basis due to the dependence on the structure of the Web page. Scrappers are useful when there is no programmatic interface to the data.

2.2 Web Services

Web Services define a mechanism for applications to programmatically exchange data and make remote procedure requests. Web Services create an application interface
that others can invoke to obtain the offered services. Web Services simplify the creation of distributed applications through the use of standards and a well defined service description language. The three areas of standards in the Web Services framework are communication protocols, service descriptions, and service discovery [24].

**Communication protocols**

Web Services require communication between the client and servers and Web Service standards specify how the programs communicate. Standards include Simple Object Access Protocol (SOAP) [31] and Representation State Transfer (REST) [33], among others. SOAP is an eXtensible Markup Language (XML)-based protocol for information exchange. There are two types of information exchange that can take place between applications using SOAP. They are messages and Remote Procedure Calls (RPC). A message exchange is when the communication happens as a SOAP request/response. A RPC exchange is when the typed values are communicated back and forth between SOAP representation and external applications [32].

The structure for messages in SOAP is an XML element with a child header element and a child body element. SOAP also specifies how to process the message and who should process it. To carry out RPC, an XML schema specification is used. This specification is used by SOAP to encode any structures data that cannot be represented in XML like data types and arguments and define the encoding in XML. The responses are decoded using the definitions in XML.

**Service Descriptions**

Web Services Descriptive Language (WSDL) describes how the service should be contacted and used. WSDL can be viewed as a guide book about the Web Service which helps any application to communicate with the Web Service. WSDL defines the abstract interface and protocol-dependant details that must be followed by the consumer of the service [24]. The abstract interface also specifies the port type, which defines the connection point to the Web Service.

The following is an example of the port type specified in WSDL of MSN Web Service,
In this example, the portType defines the input and the output for the operation “Search” and the “Action” specifies the service being contacted to carry out the operation. The connection point/endpoint supports a collection of operations/interfaces. Each of the operations defines an input message, an optional output message, and the possible interactions. The message definitions contain information about the data types that will be used during an interaction with the Web Service. The vocabulary used for the data types is standardized through the use of an XML Schema. If the communication is just a message, the binding element defines the messaging protocol and the encoding used and if it is a RPC, it specifies SOAP-RPC protocol and the encoding to be used with it.

The Web Services architecture can be seen in Figure 1. A typical Web Service X, described by its WSDL, uses SOAP to store its WSDL in the registry. A client program queries the registry using SOAP to locate the Web Service X. At this point the client reads the services SOAP definition and can then construct messages that will be exchanged with the Web Service X.

![Figure 1: Web Services Architecture](image-url)
2.3 CDIS Project

The American Red Cross, as authorized by the United States Congress in 1905, responds immediately to approximately 67,000 disasters a year. This humanitarian organization, led by volunteers but also employing full time staff people, provides shelter, food, and health and mental health services to victims of natural and man-made disasters.

There are over 800 local Red Cross chapters that provide almost daily relief services for disasters such as single family house fires to larger disasters such as floods and hurricanes. When the scope of a disaster response exceeds the capability and resources of a local chapter, Red Cross volunteers and staff from other chapters are mobilized to augment the local personnel. Local disaster workers as well as those brought in from other chapters are often unfamiliar with the resources in the affected community. The challenge is for them to know where local community services and material goods can be found. Examples include medical equipment suppliers, emergency housing, health facilities, 24-hour pharmacies, public health information, hospitals, food pantries, hotels, emergency shelters and transportation services. Although some communities have Information and Referral or ‘2-1-1’ systems, which is a service of the United Way and the Alliance of Information and Referral Systems (AIRS) similar to 911 and 411 for service listings, no uniform information system is in place to provide these workers with the information that they need to assist the victims of the disaster.

CDIS (Community Disaster Information System) was developed by Miami University to provide the relief workers with immediate access to the location of suppliers of products and services during the time of disaster. The CDIS database maintains the name, address, phone number, and other information of these suppliers. It is essential that this information is complete and up to date information of all the resources. CDIS is currently being used on a pilot basis by seven Red Cross chapters: Arlington County (VA), Atlanta (GA), Cincinnati (OH), Dayton (OH), Kansas City (MO), Orange County (CA), and San Diego (CA).

Each record in the database has the following fields: id, name, phone number, description, comments, address, city, state, last verified, last modified, source, by who, local National, zip code, twenty four hours, hours, spec contact info, fax no, email, URL,
language1, language2, language3, owner Community id and supplier confidential
information. Each record is owned by a community, which corresponds to a Red Cross
chapter or other response organization. Communities are defined as a collection of zip
codes. Each record belongs to one or more categories such as; pharmacy, hospital or
animal clinic. The categories in CDIS are organized into a hierarchical taxonomy based
upon the Alliance of Information and Referral Systems (AIRS)/Info Line Taxonomy of
Human Services. The taxonomy is used by CDIS users to index and then to locate
categories of suppliers. Figure 2 shows how a record looks in the CDIS Web portal.

Figure 2: CDIS Record

Currently, information in the CDIS database is populated and maintained
manually by Red Cross employees and volunteers. This process may be made more
efficient by automation. Our goal is to identify automated techniques using Web Services and scrapping to increase the CDIS database accuracy and completeness.

\[2.4 \text{ Related Research}\]

There has been lot of research and development in the area of Web Mining. Web mining can be viewed as the use of data mining techniques to automatically retrieve, extract and evaluate information for knowledge discovery from Web documents and services [19]. The four phases in Web mining according to Etzioni [31] are Information Retrieval (IR) (Resource Discovery), Information Selection/Extraction and Preprocessing, Generalization and Analysis. The phase that plays a major part in this research is Information Selection/Extraction and Preprocessing, the techniques used in this phase help in developing a Web-based screen scraper. The problems faced during the extraction phase according to Myllymaki [15] are finding target HTML pages on a site by following hyperlinks (navigation problem), extracting relevant pieces of data from these pages (data extraction problem), distilling the data and improving its structure (structure synthesis problem), ensuring data homogeneity (data mapping problem), and merging data from separate HTML pages (data integration problem). Few of the above mentioned problems are faced during development and the challenge is to develop a scraper that will overcome these issues.
3. Research Methodology

The hypothesis of this research is that the quantity and quality of information in the CDIS system can be enhanced by developing automated techniques for entering and maintaining the supplier information. Two sets of experiments are conducted to test the validity of this hypothesis. They are experiments to automatically

A) Populate the database
B) Maintain the database

Figure 3 shows the environment in which the experiments are conducted. The boxes to the left are Web pages that supply browser-accessible information and the boxes to the right are Web services. Data gathered using Web pages or retrieved via Web services is stored in a temporary database called Delta database. This is the automated population phase and during maintenance this database is compared with CDIS and the information in Delta database is updated which becomes potential data for CDIS.
3.1 Experiments for Automated Population

The steps involved in performing Web-based screen scraping and using Web Services for automated population of information for CDIS are discussed in this section.

Records in the CDIS database are indexed to one or more category, such as food, shelter, transportation, etc. These categories are organized in a comprehensive taxonomy licensed by AIRS/Info Line [34]. Thus, we want to determine ways to search the Web using scraping and Web services, to find suppliers of goods and services that match the existing taxonomy categories used in CDIS. We also want to locate suppliers that correspond to the geographic regions that match the Red Cross chapters in CDIS. Geographic regions are defined by counties which in turn consist of relevant zip codes.

For in automated population, zip code and taxonomy term are used to query to the Web Services and Web pages to retrieve suppliers are formed using the taxonomy term and zip code. The taxonomy term will represent the business category in which we are looking for suppliers and the zip code will represent the geographic area in which we are trying to locate these resources.

The business category names used by the Web sites and Web Services vary from the taxonomy terms used in CDIS and also from one another. For example, the table below shows the CDIS taxonomy term Baby Furniture and the corresponding business categories YellowPages.com, Yahoo Local, and Switchboard.com.

<table>
<thead>
<tr>
<th>Name</th>
<th>YellowPages</th>
<th>Yahoo</th>
<th>Switchboard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baby Furniture</td>
<td>Furniture, Children's</td>
<td>Baby Furniture</td>
<td>Children, Baby Furniture</td>
</tr>
</tbody>
</table>

Locating appropriate businesses thus requires a mapping from CDIS taxonomy terms to those used by each of the Web sites and Web Services. The details of the mappings developed are detailed in Appendix C.

The steps involved in automated population are different in Web-based screen scraping and Web Services. Each is discussed below.
3.1.1 Web-based Screen Scraping Methodology

Yellowpages.com, Superpages.com and Switchboard.com are the Web sites that were chosen to conduct experiments because the business categories used in these Web sites is similar to taxonomy used in CDIS. Yellowpages.com LLC is a joint venture operation of AT&T Inc. and BellSouth Advertising and Publishing Corporation. Switchboard.com is a wholly owned subsidiary of InfoSpace, Inc. Superpages.com is owned by Verizon.

For each of the Web sites, the HTML output from a query is analyzed to locate the tags that encompass the required information about the suppliers: Name, Address (street name, city, state and zip code) and phone number. Software programs are developed to carry out the function of querying the Web sites with the combination of different taxonomy terms and different zip codes. The program then parses the HTML to get a Document Object Model (DOM) representation in which node is the primary data type. The DOM is a language-independent standard for representing HTML or XML. The tags in HTML become nodes in the DOM tree. The DOM document tree can then be traversed to find the desired node. The information retrieved becomes potential data for the automated population of CDIS.

Data extracted from the DOM model is stored in a database with the design similar to that of CDIS. These above mentioned steps are represented as a flow diagram. The database can then used to populate (or update) the CDIS database.
3.1.2 Web Services Methodology

Msn.com [20] and Yahoo! [23] Web Services are used to conduct experiments with Web services.

Yahoo! Local is a service provided by Yahoo! to help find the names and numbers of local services and businesses in any neighborhood. It also returns the information related to them. Yahoo! requires an application id to use its Web Services and has a limit of 5000 queries per day. Yahoo! provides libraries for accessing these Web services in several programming languages including Java.

MSN Search is an information service provided by Microsoft. MSN Search Web Service enables the users to create applications in Microsoft .NET. The WSDL document describes the Web Service. Since the primary platform used in this research is Java, Apache Axis [Appendix A], which is a Java platform for creating and deploying Web
Service applications, is used to make the WSDL to Java conversion. An application id is provided by Microsoft to use the Web Service and it allows 10,000 queries per day.

The taxonomy term and the zip code combination are fed as the request field to the Web Service and the response field consists of the information about the supplier. The database is checked for duplicates and the supplier is flagged to show the status. A flow diagram (Figure 5) is drawn to understand the steps better.

![Flow diagram](image)

Figure 5: Steps involved in population using Web Services

### 3.2 Experiments for Automated Maintenance

The information collected in the Delta database from the automated population phase can be used for maintenance of CDIS. Figure 6 illustrates that the data in Delta database is compared with the data in CDIS to locate determine if suppliers already exist in CDIS. Each supplier in the Delta database is then flagged as current, new, changed or moved to show the result of the comparison with the suppliers in CDIS data, where current means the supplier is already present in CDIS and new means the supplier is not
present in CDIS, changed means there is a change in the phone number field or name field, say “CVS” and “CVS Pharmacy” and moved means there is a change in the street name field. The update on CDIS with the data from the Delta database is carried out after it is authorized by the CDIS modulator who can make a final determination to authorize the database update process.

![Automated Maintenance Diagram]

**Figure 6: Automated Maintenance**

### 3.3 Standardization Technique

One of the major issues in the current CDIS database is that the supplier records, since they were manually input by many users and do not follow any standard street address conventions. For example, an address may be entered as Main Street, Main St., Main St, or Main. This makes it difficult to automate the process of detecting duplicates and address changes for updating records. The technique used in this research to overcome this issue is to standardize each field in the record separately. This technique is
carried out in Delta database during population phase and the records from my CDIS are standardized for comparison purposes during the Maintenance phase but are not updated since the Delta will store current records in a standardized fashion. The fields that are standardized are address, state, city and phone. The address field is standardized using the USPS standard address formats (Appendix B). The city and state are related to zip code and since the zip codes that are queried are already present in CDIS, the city and state corresponding to that zip code is used in the city and state fields. The phone field is standardized to follow (XXX) XXX-XXXX format. All the entries are upper case and have the end spaces trimmed. Once the record is standardized, it is easy to compare it with another standard record for duplicate detection and for locating changes in address.

3.4 Duplicate Detection Technique

During the automated population phase there will be many supplier records retrieved that will be duplicates of records that are already present in Delta database. Detecting duplicates during the initial phase will avoid wasting time and storage space. After the retrieved record is standardized, it is then compared with the existing records in that particular zip code. Since the city and state are related to the zip code, they are always the same after standardization. If the name, address and phone number of the retrieved record matches with at least one record present in the populated database, the retrieved record is a duplicate and is ignored. Successful detection of duplicates is directly related to the success of the standardization phase. The other results such as partial matches between the two records and the strategy used to deal with them are discussed in the implementation phase.

3.5 Measures of Analysis

The success of the experiments conducted during automated population phase will be determined by measuring the number of new records inserted, percentage updated and percentage of duplicates detected. In the case of automated maintenance, the success is measured by calculating the percentage of duplicates detected and the percentage of ‘no operation’ supplier records. ‘no operation’ means that the records do not fall into any of the predefined cases and needs to be investigated by a human to decide if it needs to enter
CDIS. The number of records with status ‘new’, ‘changed’, ‘current’ and ‘moved’ are used to determine the state of CDIS with reference to Delta populated database. In both the experiments, the percentage of duplicates detected shows that the application is able to detect duplicates and ignore them efficiently and checking the database and confirming it is free of duplicates shows success of duplicate detection. The percentage of ‘no operation’ supplier records should be lower because it means that those records need to be further investigated and our system should be able to automatically maintain as many records as possible. These measures will be carried out after the experiments are implemented and a portion of it will be manually verified to test the consistency of the techniques. The following section discusses the implementation phase in detail.
4. Implementation

This section describes the experiments performed for automated population and maintenance and their results. Suppliers stored in Delta database are used for automatic population. In the maintenance phase records in the Delta database is compared with the CDIS database and the suppliers in Delta database are flagged which will be potential information for updating CDIS.

4.1 Implemented Classes

The UML class diagrams in figures 7, 8 and 9 give an overview of the software developed in this research. Figure 7 shows the relationship between the classes Supplier, ManageDups, Cases and StandardAddress. Supplier is the object that stored the basic supplier information that is retrieved and the class has accessors and modifiers for all the variables (name, city, state, address, phone). StandardAddress is the class that implements the standardization technique discussed in 3.3 and ManageDups is called when a new supplier record has to be compared with an old record for duplicate detection as discussed in 3.4. The cases are used during the automated maintenance phase to determine the case under which a record needs to be flagged.
The Class Diagram II in figure 8 shows the structure for the classes used in Web-based screen scraping. This particular figure shows the implementation of Switchboard.com and the classes for Yellowpages.com and Superpages.com follow the similar structure. The parse and print methods are used for the purpose of pattern extraction, by parsing the HTML and navigating the nodes to find necessary patterns. StandardAddress and ManageDups classes are invoked by Switchboard to achieve standardization and duplicate detection.
The class diagram for Yahoo which is shown in figure 9 depicts the implementation for a client using the Web Service. When it is compared with figure 8, we can see that it does not have the parse and print methods and the standardization and duplicate detection is carried out in this implementation as well.

![Switchboard class diagram]

---

**Figure 8: Class Diagram II**

The class diagram for Yahoo which is shown in figure 9 depicts the implementation for a client using the Web Service. When it is compared with figure 8, we can see that it does not have the parse and print methods and the standardization and duplicate detection is carried out in this implementation as well.
These class diagrams will act as a reference for the content discussed in the upcoming sections.

### 4.2 Automated Population

The steps involved in automatically populating the data and storing it in the Delta database is discussed in this section.
Figure 10: Flowchart of Automated Population

Figure 10 shows the steps in the form of the flowchart. First the zip code is selected from the CDIS database’s zip table, and then the business category for the service (Web site or Web Service) that is being queried is selected from the taxonomyTerm table in Delta database. The taxonomyTerm is pre loaded with all the mappings discussed in Appendix C. The business category and the zip code will help form the query for the service. The query will be in the form of URL in the case of Web sites and a Request object in case of Web Services. The implementation of the next step will be discussed in detail in the following sections. This is where the Response object (in
case of Web Services) or the HTML source (in case of Web sites) is analyzed and the
supplier information is extracted from it if there are suppliers present. Then the supplier
is saved to a Supplier object. The StandardAddress’s getStand() method will actually split
the City and State and will standardize the same with the help of CDIS zip table, since it
already has all the City and State names for the corresponding zip code. The following
code fragment shows the usage of the supplier object and its modifiers and the usage of
StandardAddress class.

```java
Supplier sup = new Supplier();
sup.setAdd(address);
sup.setName(name);
sup.setPhn(phno);
StandardAddress std = new StandardAddress(sup, false);
sup = std.getStand();
if (sup != null)
    checkdup(sup);
```

After the supplier is standardized, Delta database needs to be checked for
duplicate entries. The checkdup(sup) method is to check Delta database for duplicate
entries. This goes through all the suppliers in the database and compares it with the
retrieved supplier. Each supplier from the database for that zip code is taken and stored in
the Supplier object. Since every supplier in Delta database is standardized before they are
inserted, the assumption that City and State field will be the same for a given zip code
can be made. So the only other fields that need to be compared are Name, Address and
Phone. The table below shows the strategy used to identify duplicates.

<table>
<thead>
<tr>
<th>Case</th>
<th>Name</th>
<th>Address</th>
<th>Phone</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No Match</td>
<td>No Match</td>
<td>No Match</td>
<td>New entry</td>
</tr>
<tr>
<td>2</td>
<td>No Match</td>
<td>No Match</td>
<td>Match</td>
<td>New entry (same phone -- can be like two different departments)</td>
</tr>
<tr>
<td>3</td>
<td>No Match</td>
<td>Match</td>
<td>No Match</td>
<td>New entry</td>
</tr>
<tr>
<td>4</td>
<td>No Match</td>
<td>Match</td>
<td>Match</td>
<td>Update (Append name)</td>
</tr>
<tr>
<td>5</td>
<td>Match</td>
<td>No Match</td>
<td>No Match</td>
<td>New entry</td>
</tr>
<tr>
<td>6</td>
<td>Match</td>
<td>No Match</td>
<td>Match</td>
<td>New entry</td>
</tr>
<tr>
<td>7</td>
<td>Match</td>
<td>Match</td>
<td>No Match</td>
<td>Update (Append phone)</td>
</tr>
<tr>
<td>8</td>
<td>Match</td>
<td>Match</td>
<td>Match</td>
<td>Duplicate</td>
</tr>
</tbody>
</table>

Table 1: Strategy to classify retrieved suppliers

Cases 1, 2, 3, 4 and 5 classify the retrieved supplier as a new entry. In case 1 all
three fields don’t match and in case 2 same phone number can mean that it is a common
number used by many chain stores or organizations and case 3 is a new entry because on the address matches which means it can be a different business in the same location. Case 5 is a new entry because only name matches like in many chain stores and in case 6 the address is different so it is a new entry. In case 4, the name is different and the name is appended with the previous name and stored. In case 7, the phone is different so the phone numbers are appended. Whenever the case says new entry, the supplier will be inserted into the system and if it is a duplicate, the supplier is ignored and updates are carried out on the suppliers existing in the system with information from the new supplier. Every time a new supplier is entered into the database, the taxonomytermsupplier table, which associates a supplier id with the taxonomy term is updated by adding a new record which connects the supplier to the taxonomy term and when duplicates are detected, if they are under a different taxonomy term, the taxonomy term table is updated with the duplicate supplier id and the new taxonomy term.

The above procedure is carried out for every supplier retrieved during that query until no more suppliers are left. Then the next query is formed with the next business category and the same steps are repeated for the next zip code. The next section will discuss how the suppliers are extracted using each of the Web sites and Web Services.

4.2.1 Web-based Screen Scraping

The three Web sites that are used are Yellowpages.com, Superpages.com, and Switchboard.com. In all the three Web sites, the HTML source needs to be parsed to get the HTML DOM structure, and then the supplier is extracted by identifying the pattern in which it occurs in each of these Web sites. The parser used is explained followed by the information pattern in each of the Web sites.

a. Parsing the HTML

CyberNeko HTML Parser [28] scans and balances the tags in the HTML file. The DOM parser is used to parse the entire file into its DOM structure and the DOM fragment parser is used to parse fragments of the HTML document to its DOM structure. For example, a child node of a node can be parsed as a separate fragment. As said earlier, the HTML tags become nodes in the DOM and the hierarchical structure makes traversal easier.
b. Pattern Extraction

This section discusses how the information required about each supplier is organized in the HTML source in each of the Web sites.

YELLOWPAGES.COM

The information is available in a simple format in Yellowpages.com. The tag name ‘div’, with the attribute name ‘class’ and value ‘ypListingBox’ has a single business name, address and phone number and it is easy to get the name and address due to the fact that they are present in the node with attribute name ‘class’ and value ‘ypbname’ and ‘ypbaddress’ respectively. Phone number is retrieved by comparing the remaining text in the ‘ypblistingbox’ with the standard (NPA) NXX-LINE format. The page is searched for links that leads to next page containing results. Once all the pages are visited it quits or moves to the next query. A sample HTML source for pharmacy in 45056 is shown.

SUPERPAGES.COM

A single address can be scraped from the Web page by finding the attribute ‘style’ which has a value for a margin of a particular kind for the text content that has the name, address and phone number. This makes it very dependant on the structure of the Web page. When compared to Yellowpages.com, the pattern to be extracted is not explicit in
Superpages.com. The structure of the HTML where the data to be extracted is present can be seen below. Duplicate links are present for navigating to the next page which has to be ignored. The tag in which the supplier information is stored can be located by searching for its formatting information. It does not have any special attribute value like Yellowpages.com.

```
<br>(513) 523-6378</span>
```

**SWITCHBOARD.COM**

The results are traversed in two steps in Switchboard. The listing has a structure shown below.

```
The class name ‘zipcodelink’ corresponds to hyperlinked zip code. When the hyperlink given in node ‘HREF’ is followed, the Web page that is shown below is reached. By scraping this business specific page, name, address and phone number of that
```
particular business can be extracted using the attribute value containing ‘pgtitle’, ‘address’ and ‘phone’ respectively.

**CVS/pharmacy**

(513) 522-6378  
540 South Locust Street Oxford, OH 45056

Since this page has the information we need more organized it makes scraping easier. All the pages in the result set are visited to retrieve all the necessary results.

### 4.2.2 Web Services

With Web Services, the process of retrieving supplier information is very simple and it involves making use of a Request and Response object. As discussed earlier, the Web Services queries (Request object) are comprised of the business category and the zip code.

An example of the query generated using Yahoo! is shown in the following code fragment:

```java
client = new SearchClient("theseswork");

LocalSearchRequest request = new LocalSearchRequest(query);
request.setZip(zipcode);

try {
    // Execute the search.
    LocalSearchResults results = client.localSearch(request);
}
```
In this example, ‘thesiswork’ is the application ID to access Yahoo!’s Web Services, *query* corresponds to the business category being queried for and *zipcode* is the zip code being queried on.

An example for the query generated using MSN is illustrated below: the latitude 39.503838 and longitude 84.738518 correspond to the latitude and longitude of zip code 45056, this is just used as an example.

```java
MSNSearchPortType port = new MSNSearchServiceLocator().getMSNSearchPort();
SearchRequest request = new SearchRequest();

request.setAppID(\"key\");
request.setQuery(\"query\");
request.setCultureInfo(\"en-US\");
request.setSafeSearch(SafeSearchOptions.Off);
Location loc = new Location();
loc.setLocationLatitude(39.503838);
loc.setLocationLongitude(84.738518);
request.setLocation(loc);

SourceRequest singleSr =
    new SourceRequest(SourceType.PhoneBook, 1, 51, null,
                      SortByType.Distance, ResultFieldMask.All, new ArrayOfStringSearchTagFilters());

SourceRequest[] sr = {singleSr};
ArrayOfSourceRequestRequests asr = new ArrayOfSourceRequestRequests(sr);
sr.setRequest(sr);
request.setRequest(asr);

SearchResponse searchResponse = port.search(request);
```

Here ‘key’ is the application ID given by MSN, SourceType.PhoneBook will make the search a Local search, add.getname() gets the taxonomy term to be queried.

If the Web Service does not accept zip code in the Request object, the latitude and longitude of the location is given instead. The business category, zip code OR latitude and longitude are given as input to the Web Services and local search is performed. The Response object will return the list of suppliers which can be standardized and then stored in Delta database.

### 4.3 Automated Maintenance

This phase involves using the supplier information from Delta database and comparing it with the current supplier table on CDIS and maintaining the flags in Delta database which holds the potential data for CDIS. The steps involved in carrying out maintenance are shown in the Figure 11. First the zip codes are first taken from Delta database and for each zip code the suppliers present in that zip code are retrieved. Since
Delta database has supplier records from three different Web sites and two different Web Services there is a possibility that there might be an overlap in the supplier records. So the duplicates are detected before the next step. If the supplier has already been flagged then it means the current unflagged supplier taken from Delta will be a duplicate and hence deleted. For each unique supplier is compared with the suppliers present in CDIS for that particular zip code and then the supplier from Delta database is flagged with the status of that supplier with reference to CDIS. Then the next supplier is taken until all the suppliers from Delta database are compared and then the next zip code is taken and the same steps are followed.

Figure 11: Flowchart of Automated Maintenance

During the compare step, since the city and state of Delta data are standardized using the zip table in CDIS only the name, address and phone are compared. The following cases are used to flag Delta database. The flag column in Delta database will store the information on the supplier entry. If the case is ‘New’ the status will say the
same meaning the supplier is new and is absent in CDIS. The status ‘Changed’ means that one field (e.g. Phone) has been changed and ‘Moved’ means the address field has changed. The supplier is not flagged and is deleted from Delta database if the case says ‘No operation’ or ‘Match’. Since there can be many cases for a single supplier, the priorities are assigned. With finding if the record is current being the highest priority. Table 2 shows the cases in detail.

<table>
<thead>
<tr>
<th></th>
<th>Name</th>
<th>Address</th>
<th>Phone</th>
<th>Description</th>
<th>Priority (1-high)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 1</td>
<td>No Match</td>
<td>No Match</td>
<td>No Match</td>
<td>New</td>
<td>4</td>
</tr>
<tr>
<td>Case 2</td>
<td>No Match</td>
<td>No Match</td>
<td>Match</td>
<td>No Operation</td>
<td>5</td>
</tr>
<tr>
<td>Case 3</td>
<td>No Match</td>
<td>Match</td>
<td>No Match</td>
<td>No Operation</td>
<td>5</td>
</tr>
<tr>
<td>Case 4</td>
<td>No Match</td>
<td>Match</td>
<td>Match</td>
<td>Changed</td>
<td>3</td>
</tr>
<tr>
<td>Case 5</td>
<td>Match</td>
<td>No Match</td>
<td>No Match</td>
<td>New</td>
<td>4</td>
</tr>
<tr>
<td>Case 6</td>
<td>Match</td>
<td>No Match</td>
<td>Match</td>
<td>Moved</td>
<td>2</td>
</tr>
<tr>
<td>Case 7</td>
<td>Match</td>
<td>Match</td>
<td>No Match</td>
<td>Changed</td>
<td>3</td>
</tr>
<tr>
<td>Case 8</td>
<td>Match</td>
<td>Match</td>
<td>Match</td>
<td>Current</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2: Strategy to maintain suppliers
5. Results

This section summarizes the results obtained during the execution of the implemented experiments and the analytical measures discussed earlier.

5.1 Automated Population

This experiment updates the database Delta using the supplier information obtained from the Web sites using Web-based screen scraping and Web Services. The taxonomy terms [Appendix C] were used against all the zip codes in the state of Ohio for the test run. As discussed earlier (Table 1), Case 1, Case 2, Case 3, Case 5 and Case 6 correspond to new supplier records and Case 4 and Case 7 are the supplier records that need to be updated and Case 8 are duplicate records. Table 3 shows the results in detail for each Web Service and Web site used. The duplicate column shows the number of duplicates that was retrieved but ignored in each of the services during the automated population phase. The updates column shows the number of the times the suppliers were updated. The high number is due to the drawback in the system that appending names or phone numbers can cause the same record to be updated again with the name or phone number from the previous update.

<table>
<thead>
<tr>
<th>Web Site</th>
<th>Delta Database</th>
<th>Duplicates</th>
<th>Updates</th>
<th>Duplicates%</th>
<th>Updates%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yellowpages.com</td>
<td>4505</td>
<td>88155</td>
<td>5667</td>
<td>88.75%</td>
<td>6.71%</td>
</tr>
<tr>
<td>Superpages.com</td>
<td>2770</td>
<td>453</td>
<td>88</td>
<td>13.68%</td>
<td>2.66%</td>
</tr>
<tr>
<td>Switchboard.com</td>
<td>2006</td>
<td>5406</td>
<td>63</td>
<td>72.33%</td>
<td>0.34%</td>
</tr>
<tr>
<td>Yahoo</td>
<td>4298</td>
<td>24005</td>
<td>2024</td>
<td>79.15%</td>
<td>6.67%</td>
</tr>
<tr>
<td>Total</td>
<td>13578</td>
<td></td>
<td></td>
<td>63.48%</td>
<td>4.22%</td>
</tr>
</tbody>
</table>

Table 3: Automated Population analysis

The total of 13578 supplier records was inserted into Delta database. From the table we can see that % of duplicates and % updates differ between each of the Web site or Web Service used. The average percentage of duplicates detected and ignored was 63 % and the percentage of supplier records that were updated were on average 4%.
5.2 Automated Maintenance

The database (Delta) obtained after automated population is compared with the existing CDIS database. The supplier table in the CDIS database consists of 1632 records. In Table 4, the % moved, % changed and % current is with reference to the supplier records in the CDIS database and it signifies the records in CDIS in which the suppliers have moved, changed name or phone and are current. The % no operation is with reference to Delta database where there was no operation assigned to 4 records in the entire 13578. The % duplicates is the number of supplier records that overlapped in between all the Web Services and Web sites.

<table>
<thead>
<tr>
<th>CDIS OHI0</th>
<th>1632</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delta</td>
<td>13578</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cases</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>new</td>
<td>13264</td>
</tr>
<tr>
<td>changed</td>
<td>69</td>
</tr>
<tr>
<td>moved</td>
<td>6</td>
</tr>
<tr>
<td>no operation</td>
<td>4</td>
</tr>
<tr>
<td>current</td>
<td>10</td>
</tr>
<tr>
<td>duplicates</td>
<td>225</td>
</tr>
</tbody>
</table>

| % moved  | 0.368% |
| % changed| 4.228% |
| % no operation | 0.029% |
| % current | 0.613% |
| % duplicates | 1.857% |

Table 4: Automated Maintenance Results

5.3 Discussion

From the aforementioned results, the automated population phase successfully carries out the task of retrieving information from the Web using screen scraping and Web Services. It also standardizes the address and detects duplicates. This is confirmed by manually navigating the tables that store the results from automated population and it was free of duplicates and the records followed the standard format discussed in section 3.3. On an average, 63% of the retrieved records were duplicates and had there been no duplicate detection, it would be a waste of time and space. In the maintenance phase, the duplicates within the different services like Yellowpages.com, Yahoo etc. are detected.
and that constitutes 2% of the result set. The detection of suppliers who might have moved their business location or changed their contact information in the maintenance phase benefits CDIS by making the database have only the current information.

The advantages in using this approach, apart from duplicate detection and standardization are that only the module pertaining to extraction patterns has to be modified and the other modules such as standardization and duplicate detection can be reused. For example in Figure 8, the class structure will be same for any Web site while developing a screen scraper, only the print method will have to be modified according the pattern extraction technique for that Web site. Since there are separate classes for each service, we can choose which ones to run at any given time.

The disadvantages of using Web-based screen scraping are that an existing parser will fail when changes occur in the underlying HTML used by the Web site. A limitation of Web Services is query limits that restrict the particular number of queries per day. During the time of development, Switchboard.com changed the structure of the HTML which meant revising the screen scraping pattern extraction phase and the code and MSN released their next version of Search Web Service which caused the change in the WSDL file used. This required change in the experiments to accommodate the updates and invoking Axis to generate Java API for the new version of the Web Service.
6. Future Work

The future work that can be done on this project is discussed in this section. The issue with updates during the automated population phase is that the name and phone number are appended using a delimiter and every time this can be replaced by a more sophisticated method which will in turn detect and ignore more duplicates because in some cases, the duplicates are getting appended.

The taxonomytermsupplier table is not maintained during the maintenance phase, this can be a future case study. There are instances when the services retrieve incomplete supplier information. For example, the supplier might not have any phone associated with it and a future area of work can be to handle such ambiguity. Address validation which is to automatically check if the address actually exists before entering it into the database can be another interesting enhancement to this system.
7. Conclusion

The primary goal of the thesis is to develop automated techniques for entering and maintaining the supplier information to enhance the quantity and quality of information in the CDIS system.

From the results, we can see that the application is able to automatically populate and maintain the supplier records that are potential data for CDIS and it offers additional enhancements such as address standardization and duplicate detection which is currently not present in CDIS. The difficulties faced during the research were changes in structure of the Web page in Switchboard.com and changes of MSN’s WSDL both meant recoding the respective classes.

The conclusion is that using this approach, a significant portion of CDIS can be automatically populated and maintained in a standard fashion that keeps CDIS system standard, current and free of duplicates.
Appendices
Appendix A – Apache Axis

Axis is Apache’s open-source Web Services framework. In this research, Axis is used to create a Java Application Programming Interface (API) for MSN’s Web Services. The WSDL2Java class in Axis is used to do this. Web Services Descriptive Language (WSDL) describes how the service should be contacted and used. The MSN provides the WSDL file for their Web Services.

The WSDL2Java class then carries out the following operations. Each clause in the WSDL file will be converted to a Java class and the table xx shows how the type of class created varies with the type of clause.

<table>
<thead>
<tr>
<th>WSDL clause</th>
<th>Java class(es) generated</th>
</tr>
</thead>
<tbody>
<tr>
<td>For each entry in the type section</td>
<td>java class (holder if this type is used as an inout/out parameter)</td>
</tr>
<tr>
<td>For each portType</td>
<td>java interface</td>
</tr>
<tr>
<td>For each binding</td>
<td>stub class</td>
</tr>
<tr>
<td>For each service</td>
<td>service interface (service implementation (the locator))</td>
</tr>
</tbody>
</table>

Table A – WSDL2Java

Types

The types will be converted to Java classes in the example below the complexType with the name SourceResponse is converted to SourceResponse class in Java and the get set methods are declared for the objects defined in the sequence.

```xml
- <xsd:complexType name="SourceResponse">
  - <xsd:sequence>
    <xsd:element minOccurs="1" maxOccurs="1" name="Source" type="xsd:SourceType"/>
    <xsd:element minOccurs="1" maxOccurs="1" name="Offset" type="xsd:int"/>
    <xsd:element minOccurs="1" maxOccurs="1" name="Total" type="xsd:int"/>
    <xsd:element minOccurs="1" maxOccurs="1" name="Results" type="xsd:ArrayOfResultResults"/>
  </xsd:sequence>
</xsd:complexType>
```
The interface that can be used to carry out operations on the Web Service is called Service Definition Interface (SDI). The port type and the binding type of the WSDL are used to construct the SDI.

```java
private com.msn.search.soap.SourceType source;
private int offset;
private int total;
private com.msn.search.soap.ArrayOfResultResults results;

public com.msn.search.soap.SourceType getSource() {
    return source;
}

public void setSource(com.msn.search.soap.SourceType source) {
    this.source = source;
}
```

The binding class implements port type and extends the stub class. This will act as the proxy to the Web Services. It basically will do the job of converting the method calls to SOAP calls. The Stub will hide the endpoint URL, namespace, or parameter arrays which are involved in dynamic invocation via the Service and Call objects.

```xml
<wsdl:portType name="MSNSearchPortType">
    <wsdl:operation name="Search">
        <wsdl:input message="ns:SearchMessage"/>
        <wsdl:output message="ns:SearchResponseMessage"/>
    </wsdl:operation>
</wsdl:portType>
```

```java
public interface MSNSearchPortType extends java.rmi.Remote {
    public com.msn.search.soap.SearchResponse search(
        com.msn.search.soap.SearchRequest request) throws java.rmi.RemoteException;
}
```

```xml
<wsdl:binding name="MSNSearchPortBinding" type="msn:MSNSearchPortType">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    <soap:operation name="Search">
        <soap:input/>
        <soap:body use="literal"/>
    </soap:operation>
</wsdl:binding>
```

```java
public class MSNSearchPortBindingStub extends org.apache.axis.client.Stub
    implements com.msn.search.soap.MSNSearchPortType {
```
The service interface defines the get method and the locator which is the implementation of the interface will implement these methods. The locator method will locate the stub instances.

An example of how this would be invoked by a client
MSNSearchPortType port = new MSNSearchServiceLocator().getMSNSearchPort();
SearchRequest request = new SearchRequest();


**Appendix B - USPS Standard Abbreviations**

Given below are examples of the abbreviations used by United States Postal Service (USPS). Table B shows an example of the abbreviations used for State and Table C shows an example of the common suffixes used and their corresponding standard USPS abbreviations. These are used by the `StandardAddress` class to standardize addresses.

<table>
<thead>
<tr>
<th>State/Possession</th>
<th>Abbreviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALABAMA</td>
<td>AL</td>
</tr>
<tr>
<td>ALASKA</td>
<td>AK</td>
</tr>
<tr>
<td>AMERICAN SAMOA</td>
<td>AS</td>
</tr>
<tr>
<td>ARIZONA</td>
<td>AZ</td>
</tr>
<tr>
<td>ARKANSAS</td>
<td>AR</td>
</tr>
<tr>
<td>CALIFORNIA</td>
<td>CA</td>
</tr>
<tr>
<td>COLORADO</td>
<td>CO</td>
</tr>
<tr>
<td>CONNECTICUT</td>
<td>CT</td>
</tr>
<tr>
<td>DELAWARE</td>
<td>DE</td>
</tr>
<tr>
<td>DISTRICT OF COLUMBIA</td>
<td>DC</td>
</tr>
<tr>
<td>FEDERATED STATES OF</td>
<td>FM</td>
</tr>
<tr>
<td>MICRONESIA</td>
<td></td>
</tr>
<tr>
<td>FLORIDA</td>
<td>FL</td>
</tr>
</tbody>
</table>

Table B – Standard State Abbreviation

<table>
<thead>
<tr>
<th>Commonly used suffixes</th>
<th>USPS Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLEE</td>
<td>ALY</td>
</tr>
<tr>
<td>ALLEY</td>
<td>ALY</td>
</tr>
<tr>
<td>ALLY</td>
<td>ALY</td>
</tr>
<tr>
<td>ALY</td>
<td>ALY</td>
</tr>
<tr>
<td>ANEX</td>
<td>ANX</td>
</tr>
<tr>
<td>ANNEX</td>
<td>ANX</td>
</tr>
<tr>
<td>ANNX</td>
<td>ANX</td>
</tr>
<tr>
<td>ANX</td>
<td>ANX</td>
</tr>
<tr>
<td>ARC</td>
<td>ARC</td>
</tr>
<tr>
<td>ARCADE</td>
<td>ARC</td>
</tr>
<tr>
<td>AVE</td>
<td>AVE</td>
</tr>
<tr>
<td>AVE</td>
<td>AVE</td>
</tr>
<tr>
<td>AVEN</td>
<td>AVE</td>
</tr>
<tr>
<td>AVENU</td>
<td>AVE</td>
</tr>
<tr>
<td>AVENUE</td>
<td>AVE</td>
</tr>
<tr>
<td>AYN</td>
<td>AVE</td>
</tr>
<tr>
<td>AVNUE</td>
<td>AVE</td>
</tr>
</tbody>
</table>

Table C – Standard Street suffix Abbreviation
Appendix C - Category Mapping

The Web sites used in Web-based screen scraping and Web Services make use of taxonomies that are from different vendors. The categories are related to the taxonomy used by the Web site or the Web Service. These are then loaded to the TaxonomyTerm table in Delta database. Table D shows the mapping between categories in CDIS and the Web sites and Web Services used. A blank cell shows that there is no matching category in the website.

<table>
<thead>
<tr>
<th>TaxonomyTerm CDIS</th>
<th>YellowPages</th>
<th>Yahoo</th>
<th>Switchboard</th>
<th>Superpage</th>
<th>MSH Live</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blood Pressure Monitors</td>
<td></td>
<td>Medical Supplies &amp; Equipment</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bed Pans/Potty Chairs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baby Furniture</td>
<td>Furniture-Children’s</td>
<td>Baby Furniture</td>
<td>Children Baby Furniture</td>
<td>Baby Furniture</td>
<td>Childrens Furniture,</td>
</tr>
<tr>
<td>Business Disaster Loans</td>
<td></td>
<td>Business Financing</td>
<td>Disaster Loans</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baby Clothing</td>
<td>Childrens &amp; Infants Wear-Retail</td>
<td>Baby Apparel</td>
<td>Baby Clothing</td>
<td>baby clothing</td>
<td></td>
</tr>
<tr>
<td>Business Continuity Consultation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programs</td>
<td>Business Coaches &amp; Consultants</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Body Identification Coordination</td>
<td></td>
<td>Social Welfare Services</td>
<td>Social Service &amp; Welfare</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Domestic Violence Shelters</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Budget Hotels</td>
<td>Hotels, Motels &amp; Inns</td>
<td>Hotels &amp; Motels</td>
<td>Hotels &amp; Motels Reservations</td>
<td></td>
<td>Cell Phone Wireless Donation</td>
</tr>
<tr>
<td>Cell Phone Donation Programs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Braille and Tactile Aids</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bee Control</td>
<td>Bee Control &amp; Removal Service</td>
<td>Pest Control</td>
<td>Bee Control &amp; Removal Service</td>
<td>Bee Control</td>
<td>Bee &amp; Wasp Removal</td>
</tr>
<tr>
<td>Biomedical Engineers</td>
<td>Engineers-Professional-Biomedical</td>
<td>Environmental Consulting</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bulldozer Operator Volunteer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Opportunities</td>
<td>Volunteer Workers Placement Service</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blind Mobility Aids</td>
<td></td>
<td>Blind &amp; Vision Impaired Services</td>
<td>Blind Mobility Aids</td>
<td>Blind Mobility Aids</td>
<td>Blind and Low Vision Products</td>
</tr>
<tr>
<td>AmeriCorps VISTA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Backpacks</td>
<td>Contractors Equipment &amp; Supplies</td>
<td>Excavation Work</td>
<td>Excavating Contractors</td>
<td>Backpacks</td>
<td>Backpacks</td>
</tr>
<tr>
<td>Bank/Saving Suites</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bird Control</td>
<td></td>
<td>Bird Barriers, Replants &amp; Controls</td>
<td>Bird Barriers, Replants &amp; Controls</td>
<td>Bird Control</td>
<td>Bird Control</td>
</tr>
<tr>
<td>Children &amp; Clothing</td>
<td>Childrens &amp; Infants Wear-Retail</td>
<td>Baby Apparel</td>
<td>Childrens &amp; Infants Wear</td>
<td>Children’s Clothing</td>
<td>Children’s Clothing</td>
</tr>
<tr>
<td>Chemical Engineers</td>
<td>Engineers-Professional-Chemical</td>
<td>Science Engineers</td>
<td>Engineers Chemical</td>
<td>Chemical Engineers</td>
<td>Chemical Engineers</td>
</tr>
<tr>
<td>Beds</td>
<td>Beds &amp; Headboards</td>
<td>Beds &amp; Mattresses</td>
<td>Beds &amp; Accessories Retail</td>
<td>Beds</td>
<td>Beds</td>
</tr>
<tr>
<td>Court Closures/Reassignments</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Civil Engineers</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Child Care Centers</td>
<td></td>
<td>Childcare Services</td>
<td>Child Care Service</td>
<td>Child Care Centers</td>
<td>Child Care Centers</td>
</tr>
<tr>
<td>Elder Abuse Shelters</td>
<td></td>
<td>Social Welfare Services</td>
<td>Elder Abuse</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table D – Category Mapping

Figure A shows the view after it is loaded into the taxonomyTerm table in the database.
<table>
<thead>
<tr>
<th>#</th>
<th>Term</th>
<th>Category</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>Blood Pressure Monitors</td>
<td>YellowPages</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Bed Frames/Sheets</td>
<td>Furniture-Children's</td>
<td>Programs the</td>
</tr>
<tr>
<td>25</td>
<td>Baby Furniture</td>
<td>Furniture-Children's</td>
<td>Programs the</td>
</tr>
<tr>
<td>32</td>
<td>Business Disaster Loans</td>
<td>Business Training</td>
<td>Programs the</td>
</tr>
<tr>
<td>50</td>
<td>Baby Clothing</td>
<td>Children &amp; Infants</td>
<td>Programs the</td>
</tr>
<tr>
<td>74</td>
<td>Business Continuity Consultation</td>
<td>Business Continuity</td>
<td>Programs the</td>
</tr>
<tr>
<td>10</td>
<td>Body Identification Orientation</td>
<td>Body Identification</td>
<td>Programs the</td>
</tr>
<tr>
<td>2</td>
<td>Domestic Violence Shelters</td>
<td>Domestic Violence Shelters</td>
<td>Programs the</td>
</tr>
<tr>
<td>18</td>
<td>Budget Hotels</td>
<td>Social Service &amp; Welfare</td>
<td>Programs the</td>
</tr>
<tr>
<td>20</td>
<td>Cell Phone Donation Program</td>
<td>Cell Phone Donation</td>
<td>Programs the</td>
</tr>
<tr>
<td>56</td>
<td>Braille and Tactile Aids</td>
<td>Tactile Aids</td>
<td>Programs the</td>
</tr>
<tr>
<td>56</td>
<td>Bee Control</td>
<td>Bee Control &amp; Removal</td>
<td>Programs the</td>
</tr>
<tr>
<td>46</td>
<td>Biomedical Engineers</td>
<td>Biomedical Engineering</td>
<td>Programs the</td>
</tr>
<tr>
<td>47</td>
<td>Bulb/Photo/Book/Volunteer Work/Visitor</td>
<td>Environmental Concerns</td>
<td>Programs the</td>
</tr>
<tr>
<td>89</td>
<td>Blind Mobility Aids</td>
<td>Blind Mobility Aids</td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>Biomedical Education</td>
<td>Biomedical Education</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>Backboards</td>
<td>Excavation Contractor</td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>Bomber/Rescue</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>95</td>
<td>Bird Control</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Children's Clothing</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Chemical Engineers</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>Beds</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>Court Closure/Remanagement</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Civil Engineers</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Child Care Centers</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>97</td>
<td>Elder Abuse Shelters</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Federal Communications</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>Clothes Driers</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Breathing Apparatus Support Units</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Community Disaster Education</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>Emergency Bureau</td>
<td>Excavation Worker</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>Emergency Medical Technician</td>
<td>Excavation Worker</td>
<td></td>
</tr>
</tbody>
</table>

Figure A – TaxonomyTerm table
Bibliography


Available: http://graphics.lcs.mit.edu/~rcm
20. Microsoft Live Development Center
21. Introduction to Web Services,
22. YellowPages,
23. Yahoo! Local,
24. IBM Web Services tutorial, “Web Services - the Web's next revolution”
25. “Web Services Conceptual Architecture”, IBM
26. Switchboard, InfoSpace, Inc
27. Superpages, Verizon

28. CyberCyberNeko HTML Parser

29. Apache’s Axis

30. United States Postal Service

31. World Wide Web Consortium, W3C


34. AIRS/211 LA County Taxonomy