Implementing a Dynamic Web Directory in Java  
Class Project, SWE 642-003, Spring 2008  
Instructor: Nick Duan, nduan@gmu.edu  
January 29, 2008

Objective:

- To understand the various design concepts of Web engineering via the development of a dynamic Web Directory service in Java.
- To obtain hands-on experience with various techniques, including HTML, JavaScripts, Java Servlet, JavaServer Pages, XML/RDF, JDBC, web crawling, and index engines, and to enhance knowledge and experience with real-world software development in a team environment.

Overview:

The project is to implement a Web Directory service that provides catalog information of content and services offered by various web sites. The Web Directory is a combination of an Internet search engine and a content categorization service. The search engine crawls the web for catalog information (or metadata about the content and services) presented by content and/or service providers in standard format. The service providers and/or content offers specify the catalog information in a standard XML file, and posted it as `siteinfo.xml` on their own web sites. A corresponding `siteinfo.html` may also be published on the offer’s/provider’s web site to display user-friendly catalog info, via XSLT translation of the `siteinfo.xml` file. The catalog information in `siteinfo.xml` from web sites will be consolidated and compiled to create one or more directories of web content and/or services. The directory information on the Web Directory will be used for directory-based, category-oriented search that will increase the accuracy of search results by narrowing the search space.

The standard catalog includes the author/company information, channel type, initial creation date and update date, version info, content category and subcategory information, geospatial code, and expiration date. The channel type consists of a set of predefined types such as web pages, web services, RSS and other content delivery types offered by the web site. The content category and subcategory are basically the taxonomical information for categorizing the content or service types. The value of category or subcategory is defined as the directory hierarchy specified in the Open Directory Project. The entire directory structure is specified in RDF format and freely downloadable from [http://rdf.dmoz.org](http://rdf.dmoz.org).

The basic architecture of the Web Directory service is depicted in the following diagram:
Requirement:

1. The Web Directory shall be implemented as a web application running on Apache Tomcat, with a search component for providing search GUI, and a directory component for displaying categorized catalog information of web sites. The search and discovery engine is provided by Apache Nutch, which is built on top of Lucene, running on Tomcat. A relational database shall be used to store the catalog content of the web sites for creating the directory.

2. Search Component: The search component should provide a GUI for user to enter textual search queries. It shall take category information provided by the directory as part of the search criteria.

3. Directory Component: The directory component shall display the directory information according to its hierarchical structure. The catalog data are defined as the leaves of the hierarchy. The GUI may have similar views as shown on the web site of Open Directory Project (http://www.dmoz.org).

4. Web Crawler: The web crawler is provided by Apache Nutch, an open source search engine package based on the highly successful Apache Lucene. The robot API of Nutch may be extended to support parsing and processing of the siteinfo.xml files. At the same time, if dealing with content providers, the web content shall be indexed by Nutch’s indexer.

5. The entire application shall be developed in a 3-tier architecture, with the presentation, business and database layer representing each of the 3 tiers, respectively. The presentation layer shall be developed as a combination of HTML, JavaScripts and/or JSPs. All string and number validation of input fields should be done via JavaScripts functions. The business tier is composed of a set of Java Servlets, and use JDBC API to communicate with the back-end database.
6. The Web Directory application shall be equipped with the capability of basic authentication, alone with a simple admin GUI for managing the user accounts.

7. It is assumed that each web site visited by the web crawler of the Web Directory is equipped with the catalog information defined in a siteinfo.xml file on the content-root path. The XML schema of the siteinfo.xml file will be provided by the instructor. A proxy server may be constructed as part of this project to mimic the behavior of each web site providing its catalog information.

8. The Web Directory shall use the standard taxonomy used by the Open Directory Project for defining its directory structure. The ODP’s directory definition is specified in a RDF file downloadable from http://rdf.dmoz.org.

9. The relational database is used for storing the catalog information from the web sites. The project team is responsible for creating its own database schema design.

10. The project shall be completed individually or by a group of up to four students. If working in a group, each student will receive the same final score for the project after delivery. The project due date is April 30, 2008.

Deliverables:

1. A design document containing the following contents (10 pages approximately. The contents except the cover page should be delivered in HTML format accessible from your personal web page on the department UNIX machine):
   - Cover page, with your name and student number. In case of a group, clearly indicates the group leader’s name and number. The URL of the design document and the URL of the application, along with the administrator’s username and password. A hard copy of this page is to be submitted on the project due day.
   - Overview of your application, e.g. what it does and why it is good (be a salesman/woman)
   - Web page design, with a brief description of each page and technology used, and a general navigation chart of all pages.
   - Servlet/JSP/JavaScript design, with description of each component/class and its invocation path, along with interfaces to access Apache Nutch.
   - Summary of your development process and ideas for future enhancement (use your imagination and creativity).
   - Database schema design and
   - Complete API documentation of all your Java classes, generated by javadoc. It must be accessible from your web page.

2. A project package including a J2EE war file containing the application, the design document in HTML forms (including JavaDocs), and Java source code, delivered on your class account on project due date. Proper coding and comment standard should be followed for writing the Java code. In case of a group, the lead person’s account shall be used for delivering the final package.

3. One or more SQL compliant database DDL files for created the project database. Proper comments shall be included in the DDL files.

Grading policy:
<table>
<thead>
<tr>
<th>Functionalities</th>
<th>Percentage of project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design document and project standards</td>
<td>10%</td>
</tr>
<tr>
<td>Directory Component</td>
<td>20%</td>
</tr>
<tr>
<td>Search Component</td>
<td>15%</td>
</tr>
<tr>
<td>Indexer</td>
<td>10%</td>
</tr>
<tr>
<td>Web Crawler</td>
<td>15%</td>
</tr>
<tr>
<td>Security</td>
<td>15%</td>
</tr>
<tr>
<td>Database Design and Implementation</td>
<td>10%</td>
</tr>
<tr>
<td>Usability</td>
<td>5%</td>
</tr>
</tbody>
</table>

**Suggested Development Steps:**

1. It is recommended that you follow the standard software development process, i.e. from analysis to design, then to implementation and testing. A good start would be trying to understand the concept of Web crawler, indexer, and Web components and their APIs, and to layout your web pages and their relationship to servlets/JSPs, Java class hierarchies and relationships. Don’t rush to implementation. A good design would make your implementation much easier.

2. You may want to divide the application into modular parts, e.g. the directory component, the search component, the web crawler, indexer, and the database component. Once you have a thorough design completed and the interface among these modules defined, you may proceed to implement and test each module one at a time, without having them interact with each other. Once you have each module fully implemented and tested, you can proceed with the integration. This is especially true if you have three people in a group and you want to have one person responsible for one or two module. This would be also a good opportunity to develop your teamwork skills.

3. It is recommended to complete your design, and then start implementing your system based on your design (not vice versa). If anything you found out during the implementation stage that something is wrong with your design, go back to your documentation and correct it before continuing with the implementation. Update your design document as you redesign and code.

4. For application development and testing, each student/group is encouraged to install and use J2SE and Tomcat on his/her personal machine, and may use MS access or any other JDBC-enabled personal databases.
Background Info - Why Web Directory?

The search and retrieval of relevant information on the Web becomes a real challenge nowadays as more and more data and content providers overflowing the Internet. Internet users depend primarily on mega search engines such as Google and Yahoo for search of information. The search engines of Google and Yahoo rely on creating huge indices of vast web content, and use key word analysis to narrow search space and provide relevance ranking of results. The accuracy and relevance ratings of the search results depend on the generality of the search terms and the heuristic information on the search terms/phrases the search engines have accumulated. It is not uncommon that the search engine returns some irrelevant results or ranks the relevance of the results inappropriately.

The general concept of increasing the search accuracy is to narrow the search space, such as providing additional conditional phrases to the search query. A more sophisticated approach is to use semantic understanding to guide searches and establish a semantic-friendly search space based on a standard ontological framework. A simple framework would be an Internet directory of web sites that can be used for narrowing search space of a given query to a subset of the whole web content space. A directory is basically a categorization scheme based on some taxonomy. Taxonomy is usually considered as a subset or an initial form of ontology.

There are many directory sites on the Internet. Many of them are targeting specific vertical markets. The directories on the Internet are usually constructed manually by editors. Yahoo, for instance, has an army of site editors to classify web site and web pages into content categories. The Open Directory Project (http://www.dmoz.org) relies on tens of thousands volunteer editors to inspect various web sites before adding the web site entries to the directory. The process of adding a new entry may take more than two weeks, and there is no way to check if the site entries are up-to-date or not. Therefore, it is highly desirable to establish an automated way for adding site entries to the directory in a non-manual fashion, and keeping the directory information up-to-date constantly.

The common consensus is that web crawler with indexed search engine (e.g. Google) works much better than directory-based search approach (e.g. Yahoo), because of advanced natural language processing and textual indexing techniques. However, as the search space grows exponentially on the Internet, some method of divide-and-concur is necessary to reduce the search space and improve the search accuracy. One common way to reduce the search space is to define a taxonomy for categorize content and data sources so that more targeted search can be conducted. This requires that category information to be defined at the content and data source level with the right granularity.

Considering that the data sources and web content are offered through web sites, it is conceivable that, instead of having the site entries in the directory created manually, the web content publishers or the service providers can define the metadata about the content and services, e.g. catalog information, in a designated location on the web server in standard format such that they can be discovered by a web crawler for generating the web directory automatically. Site information can be created and modified by site administrators dynamically, and the changes can be reflected automatically on the web
directory. This would eliminate the need of using site editors by the web directory, and the content category information is expected to be more accurate since it is provided by the content providers. One of the key reasons that directory-based search sites (e.g. Yahoo) didn’t work well is that there was a single taxonomy available to all content providers. This makes categorization extremely difficult since some content may belong to multiple categories. For instance, if a web site is offering educational robots, it could belong to both an education related taxonomy, or a technology/artificial intelligence related taxonomy. With a flexible catalog data structure, it is possible to assign multiple taxonomies to web content site or data source, so that data content won’t be locked to a single taxonomical path.

**What are the potential applications of a Web directory and what are the benefits?**

There are many potential applications that a Web directory can enable. In addition to directory-based search with increased search accuracy, a web directory can be used to support vertical search engines, metadata-driven/federated search, dynamic service directory, and collaborative creation of semantic web.

**Vertical Search Engines**

Vertical search engines are search engines tailored to specific vertical markets, such as K12 education, agriculture, manufacturing, construction, etc. Research indicates that more consumers and Internet users are drawn to vertical search sites as those sites become more mature in their own special niche market. Advertising dollars are also moving gradually from big mega search engine sites to vertical search engine sites, as more advertisers are realizing the benefits and cost-effectiveness of targeted marketing.

Vertical search sites can be easily built with support of a web directory that is powered by some taxonomy specific to that vertical market, since such a web directory is generated dynamically instead of manually. The market-specific web directories can also help consumers to navigate the Web in a structured and informed way. For instance, a parent who wants to protect his/her children from getting indecent content on the Web, could restrict the browser navigation only to web sites listed in a K12 education directory and some cultural, history, or technology directories. The applications that would spawn off from vertical search directories will be endless.

**Metadata-driven Federated Search**

By embedding the taxonomy and other metadata information (e.g. author/publisher, geospatial, temporal, legal, data format, content type, and security/access control, etc.) in the catalog file on each content site or data sources, a web directory can be used to support metadata-driven federated searches. A federated search engine is a mega search engine which does not maintain content indices but routes search queries to other search engines. It differs from a directory-based search engine in the use of a centralized index repository or distributed index repositories. A federated search engine has the advantages of performing targeted search based on taxonomical and/or ontological information of query terms, and it is especially useful in a search environment, such as the intelligence community, where the creation of a common index repository is not possible.
Dynamic Service Directory

The catalog information is not restricted to contain only metadata of web content, but could also contain information about services such as Web services or any dynamic web applications (e.g. RSS, Atom, Blog, etc.). A web directory can be easily turned into a service directory, where service information can be dynamically discovered, published, and validated. The existing service directory technologies, such as UDDI and ebXML, rely on the registry mechanism where services have to be manually registered with a central registry. This often causes inconsistencies of service definitions in the registry and the actual services hosted on remote web servers. By having the service information published right at the sites where the services are hosted, not only eliminates the inconsistencies of service information, but also allows dynamic categorization of services to enable better usability and governance control. Since the catalog can be customized easily to include secure access information, a web directory can be used as a naming service for retrieving authorization information to achieve cross-domain/enclave access control.

Collaborative Creation of a True Semantic Web

The current effort in semantic web is limited to the construction of some RDF/ontological resources whose content is usually centralized and created manually. For instance, if someone wanted to create some ontology, he or she would use some OWL editing tool to create some OWL data files, going through one or more iterations to improve it, going through some quality assurance process before releasing it. There is lack of collaboration and coordination among various web content owners and publishers. If there is framework available on the web that support interactions and collaborations among the content providers, as well as consumers, the creation of a true semantic web would become easier. Wikipedia, for instance, is such a framework for people authoring and sharing knowledge on the web in a collaborative way. It has a mechanism for creating content and content categories dynamically. By having each content provider define his/her own content catalog description, and establishing a standard for creating web directories dynamically, a framework for linking web resources and their semantic relationships can be established.

The real ultimate goal of Web directory is to create a self-evolving eco-environment for fostering the dynamic creation of a semantic web. Web directories will be the infrastructure backbone for this eco-environment by classify web content into proper categorizations for easy search and retrieval. Someday, there could be hundreds of web directory on the Internet, created dynamically instead of manually. Each is focusing on some niche markets and providing value-added, market-specific content and services to consumers and users.