Web Service Architecture for a Meta Search Engine

K.Srinivas
Associate Professor, Department of IT, Geethanjali College of Engineering & Technology, Cheeryal(V), Keesara(M), Ranga Reddy, Andhra Pradesh-501301, India.

P.V.S. Srinivas
Professor & Head, Department of CSE, Geethanjali College of Engineering & Technology, Cheeryal(V), Keesara(M), Ranga Reddy, Andhra Pradesh-501301,India.

A.Govardhan
Professor, Department of CSE, School Of Information Technology, Jawaharlal Nehru Technological University Hyderabad, Kukapally, Hyderabad,
Andhra Pradesh-500085, India.

***Abstract*—With the rapid advancements in Information Technology, Information Retrieval on Internet is gaining its importance day by day. Nowadays there are millions of Websites and billions of homepages available on the Internet. Search Engines are the essential tools for the purpose of retrieving the required information from the Web. But the existing search engines have many problems such as not having wide scope, imbalance in accessing the sites etc. So, the effectiveness of a search engine plays a vital role. Meta search engines are such systems that can provide effective information by accessing multiple existing search engines such as Dog Pile, Meta Crawler etc, but most of them cannot successfully operate on heterogeneous and fully dynamic web environment. In this paper we propose a Web Service Architecture for Meta Search Engine to cater the need of heterogeneous and dynamic web environment. The objective of our proposal is to exploit most of the features offered by Web Services through the implementation of a Web Service Meta Search Engine.**

Keywords-Meta Search Engine; Search engine; Web Services.

I. Introduction

A Meta Search Engine is a search tool that sends user requests to several other search engines and/or databases and aggregates the results into a single list or displays them according to their source. Meta Search Engines enable users to enter search criteria once and access several search engines simultaneously [18]. More comprehensive search results can be obtained by combining the results from several search engines. This may save the user to use multiple search engines separately.

This paper explores the web services and also the Architecture of a Meta Search Engine. In section 2, we propose the existing background and architecture of a Meta Search Engine. Section 3 describes the architecture of a Web Service framework. Section 4 describes about the Meta Search Engine architecture, components and function of the each component. In section 5, we propose the Web Service Meta Search Engine architecture and Section 7 provides the advantages of Web Service Metasearch Engines. Section 8 finally presents the conclusions and the future work.

II. Background

There are many methods for finding information, but one of the leading ways is through search engines. Now a day, everyone uses search engines for research, school, business, shopping, or entertainment. So, the traffic on the Web is growing exponentially [2]. To understand the working strategy of a Search Engine, one should have an overview and clear understanding of Information Retrieval System. Information retrieval deals with the representation, storage, organization and access to information items in order to give the user desired information. A distinction between traditional information retrieval and web information retrieval is that in traditional or classic information retrieval, the process of search is simple[6] and these document collections are stored in physical form. An example would be looking for information in books of a public library. Nevertheless, nowadays, most of the documents are computerized that can be retrieved with the help of a computer. Web information retrieval is on the other hand, not like the traditional IR, where, searching is performed within the globally largest collection of documents that are linked, such as the well known search services on the internet like Google or Yahoo [4].

Intense stress on user requirements recommended the architecture of Meta Search Engine. A few Meta Search Engines has been already proposed that provides quick response with re ranked results after extracting user preference. It uses Naïve Bayesian Classifier for re ranking. An enhanced version of open source Helios Meta Search Engine takes input keywords along with specified context or language and gives refined results as per user's need [8]. All the proposed solutions refine search-results up to some extent but they have a serious drawback, which is that the user profile is not stationary. The idea behind metasearch is to use multiple "helper" search engines to do the search, then to combine the results from these engines. Engines that use metasearch include Metacrawler, SavvySearch, MSN Search and Altavista, among others [11].

III. Web Services

Web service protocol is designed for providing service via web. Web Services are emerging as the fundamental building blocks for creating distributed, integrated and interoperable solutions across the Internet. They represent a new paradigm in distributed computing that allow applications to be created from multiple Web Services dispersed across the web originating from various sources regardless of where they reside or how they were implemented [19]. Unlike services in general, Web services are based on specifications for data transfer, method invocation and publishing. This is often misunderstood and when a Web service is mentioned it sometimes refers to a general service provided on the Web, like the weather forecast on a Web page for example. The weather forecast is a service and provides its functionality for a variety of users but unless it comprises an interface to communicate with other applications via SOAP [16]. Web services can be seen as software components with an interface to communicate with other software components. They have a certain functionality that is available through a special kind of Remote Procedure Call. In fact they even evolved from traditional Remote Procedure Calls.There are various aspects of Web services. Messaging, discovery, portals, roles, and coordination. The format of the messages exchanged between a client and a Web service is specified by a standard called SOAP. The Simple Object Access Protocol (SOAP) is defined by the W3C as a lightweight protocol for exchange of information in a decentralized, distributed environment. SOAP is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined data types, and a convention for representing remote procedure calls and responses [12],[16].

Web service messages are sent across the network in an XML format defined by the W3C SOAP specification. In most Web services, there are two types of SOAP messages: requests and responses. When a SOAP request is received, the Web service performs an action based on the request and returns a SOAP response. In many implementations, SOAP requests are similar to function calls with SOAP responses returning the results of the function call [12]. With SOAP, a communication between Web services is possible and structured and each participant knows how to send or receive the corresponding SOAP Message. The final step to complete the communication architecture of Web services is to define how to access a service once it is implemented.

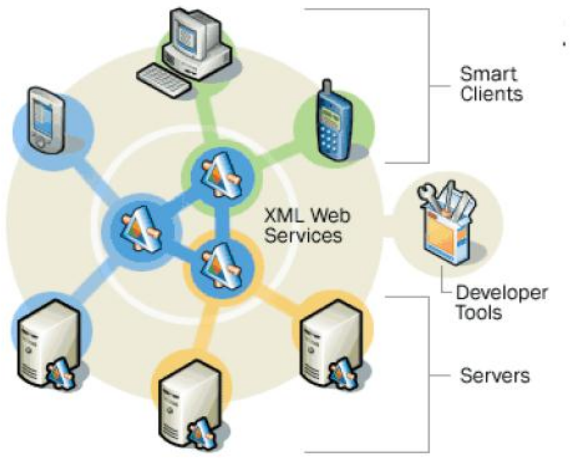

Figure 1.

Web services are the core any type of service, as it holds and connects everything together as shown in Figure 1.

IV. META SEARCH ENGINE

Meta Search Engines can be classified into two types. a) General purpose metasearch engine and b) Special purpose Meta Search Engines. The former aims to search the entire Web, while the latter focuses on searching information in a particular domain (e.g., news, jobs).

- Major Search Engine Approach**:** This approach uses a small number of popular major search engines to build a metasearch engine. Thus, to build a general-purpose metasearch engine using this approach, we can use a small number of major search engines such as Google, Yahoo!, Bing (MSN) and Ask. Similarly, to build a special purpose Meta Search Engine for a given domain, we can use a small number of major search engines in that domain.

- Large scale Metasearch engine approach**:** In this approach, a large number of mostly small search engines are used to build a Metasearch engine. For example, to build a general-purpose metasearch engine using this approach, we can perceivably utilize all documents driven search engines on the Web. Such a metasearch engine will have millions of component search engines. Similarly to build a special purpose metasearch engine for a given domain with this approach, we can connect to all the search engines in that domain. For instance, for the news domain, tens of thousands of newspaper and news-site search engines can be used.

Each of the above two approaches has its advantages and disadvantages. An obvious advantage of the major search engine approach is that such a metasearch engine is much easier to build compared to the large-scale metasearch engine approach because the former only requires the metasearch engine to interact with a small number of search engines. Almost all currently popular metasearch engines, such as Dogpile, Mamma and MetaCrowler, are built using the major search engine approach, and most of them use only a handful of major search engine. One example of a large-scale special-purpose metasearch engine is AllInOneNews, which uses about 1,800 news search engines from about 200 countries/regions. In general, more advanced technologies are required to build large-scale metasearch engines. As these technologies become more mature, more large-scale metasearch engines are likely to be built.

V. PROPOSED ARCHITECTURE

The proposed architecture takes both approaches into consideration for designing web service metasearch engine system. Significant software components included in this architecture search engine selector, search engine connectors, result extractors and result merger.

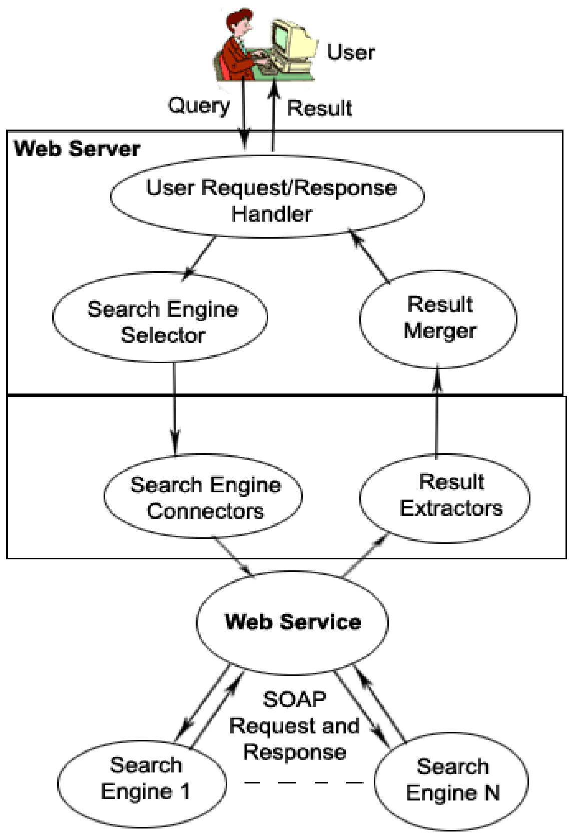

Figure 2 Web Service Meta Search Engine Architecture

A. Search Engine Selector:

If the number of component search engines in a metasearch engine is very small, say less than 10, it might be reasonable to send each user query submitted to the metasearch engine to all the component search engines. In this case, the search engine selector is probably not needed. However, if the number of component search engines is large, as in the large scale Meta Search Engine scenario, then sending each query to all component search engines will be an inefficient strategy because most component search engines will be useless with respect to any particular query. For example, suppose a user wants to find 50 best matching results for his/her query from a metasearch engine with 1,000 component search engines. Since the 50 best results will be contained in no more than 50 component search engines, it is clear that at least 950 component search engines are useless for this particular query.

Passing a query to useless search engines may cause serious problems for efficiency. Generally, sending a query to useless search engines will cause waste of resources to the metasearch engine server, each of the involved search engine servers and the Internet. Specifically, dispatching a query, including needed query reformatting, to a useless search engine and handling the returned results, including receiving the returned response pages, extracting the result records from these pages, and determining whether they should be included in the final merged result list and where they should be ranked in the merged result list if they are to be included, waste the resources of the metasearch engine server; receiving the query from the metasearch engine, evaluating the query, and returning the results back to the metasearch engine waste the resources of each search engine whose results end up useless; and finally transmitting a query from the metasearch engine to useless search engines and transmitting useless retrieved results from these search engines to the metasearch engine waste the network resources of the Internet. Therefore, it is important to send each user query to only potentially useful search engines for processing.

The problem of identifying potentially useful component search engines to invoke for a given query is the search engine selection problem, which is sometimes also referred to as database selection problem, server selection problem, or query routing problem. Obviously, for metasearch engines with more component search engines and/or more diverse component search engines, having an effective search engine selector is more important.

B. Search Engine Connectors:

After a component search engine has been selected to participate in the processing of a user query, the search engine connector established a connection with the server of the search engine and passes the query to it. Different search engines usually have different connection parameters. As a result, a separate connector is created for each search engine. In general, the connector for a search engine S needs to know the HTTP (Hyper Text Transfer Protocol) connection parameters supported by S. There are three basic parameters, (a) the name and location of the search engine server, (b) the HTTP request method (usually it is either GET or POST) supported by S, and (c) the name of the string variable that is used to hold the actual query string.

When implementing metasearch engines with a small number of component search engines, experienced developers can manually write the connector for each search engine. However, for large-scale metasearch engines, this can be very time-consuming and expensive. Thus, it is important to develop the capability of generating connectors automatically.

An intelligent metasearch engine may modify a query it receives from a user before passing it to the search engine connector if such a modification can potentially improve the search effectiveness. For example, a query expansion technique may be used by the metasearch engine to add terms that are related to the original user query to improve the chance for retrieving more relevant documents.

C. Web Service:

A Meta search engine is a search engine that collects results from other search engine. Web service offers such functionality and then presents a summary of that information as the results of a search. Most search engines available on the Web provide only a browser based interface; however, because Web services start to be successful, some of those search engines offer also an access to their information through Web services. Two types of search engines are observed, one that acts like a wrapper for the HTML pages returned by the search engine and other one is build upon the Web service offered by the search engine but this difference is visible only when looking at the internal processing of the service. It is difficult to distinguish them from the outside as they implement the same interface.

Web service are built for, any process that can be integrated into external systems through valid XML documents over Internet protocols. This definition outlines the general idea of Web services. Web services can be seen as software components with an interface to communicate with other software components. They have a certain functionality that is available through a special kind of Remote Procedure Call.

SOAP, the Simple Object Access Protocol [16] was developed to enable a communication between Web services. It was designed as a lightweight protocol for exchange of information in a decentralized, distributed environment. SOAP is an extensible, text-based framework for enabling communication between diverse parties that have no prior knowledge of each other. This is the requirement a transport protocol for Web services has to fulfill. SOAP species a mechanism to perform remote procedure calls and therefore removes the requirement that two systems must run on the same platform or be written in the same programming language.

D. Result Extractors:

After a component search engine processes a query, the search engine will return one or more response pages. A typical response page contains multiple (usually 10) search result records, each of which corresponds to a retrieved Web page, and it typically contains the URL and the title of the page, a short summary (called snippet) of the page content, and some other pieces of information such as the page size. Fig 2 shows the upper portion of a response page from the Google search engine. Response pages are dynamically generated HTML documents, and they often also contain content unrelated to the user query such as advertisements (sponsored links) and information about the host Web site.

A program (i.e., result extractor) is needed to extract the correct search result records from different component search engines can be merged into a single ranked list. This program is sometimes called an extraction wrapper. Since different search engines often format their results differently, a separate result extractor is usually needed for each component search engine. Although experienced programmers can write the extractors manually, for large-scale metasearch engines, it is desirable to develop techniques that can generate the extractors automatically.

E. Result Merger:

After the results from the selected component search engines are returned to the metasearch engine, the result merger combines the results into s single ranked list. The ranked list of search result records is then presented to the user, possible 10 records on each page at a time, just like more search engines do. Many factors may influence how result merging will be performed and what the outcome will look like. The information that could be utilized includes the local rank of a result record from a component search engine, the title and the snippet of a result record, the full document of each result, the publication time of each retrieved document, the potential relevance of the search engine with respect to the query from where a result is retrieved, and more. A good result merger should rank all returned results in descending order of their desirability.

The existing architecture has many disadvantages in search engine selection, search engine connection and result extractors. We proposed a robust metasearch engine architecture using web services for heterogeneous and dynamic environment.

VI. Implementation

WSMSE is a web based Meta Search Engine that is developed using Java Web Service, Servlet and JSP. This search engine is developed based on the assumption that an average web user makes searches based on imprecise query keywords or sentences, which in turn leads to unnecessary or inaccurate results. In this work, we demonstrate that with a simple tweak or manipulation of existing search engine functions, which can helps users to get better search results.

To build and deploy Web service Java Web Service Developer Pack (WSDP) will be used. Based on the Java 2 SDK, this toolset adds new API including XML Messaging (JAXM), XML Processing (JAXP), XML Registries (JAXR), XML-based RPC (JAX-RPC) and the SOAP with Attachments (SAAJ). The main advantage Java WSDP is it supported heterogeneous platform and have dynamic behavior.

Every Web service is deployed with a Web Service Description Language (WSDL) file that acts like an instruction manual; in case someone wants to use a particular Web Service, he simply has to look at that WSDL file to learn how to communicate with the corresponding service and use it. Amongst all advantages, the most important one for the Web services is that XML is not platform dependent and it allows easy data processing and exchange between different applications.

HTTP is the most popular transfer protocol and it’s supported on almost all platforms. By implementing this standard, combined with XML, Web services remove almost all frontiers between platforms.

The Simple Object Access Protocol (SOAP) exchange of information in a decentralized, distributed environment. SOAP is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined data types, and a convention for representing remote procedure calls and responses.

```
<?xml version="1.0" encoding="utf-8"?>

<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelop
e/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/enc
oding/"                    xmlns:tns="http://adv/WSMSE"
xmlns:types="http://adv/WSMSE/encodedTypes"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<soap:Body
soap:encodingStyle="http://schemas.xmlsoap.org/soap
/encoding/">

<tns:GetAvailableSEWS />

</soap:Body>

</soap:Envelope>
```

Figure 3 SOAP request to the WSMSE, invoking the Get Available Search Engine Web Service

Web Service Description Language (WSDL) defines an XML grammar for describing network services as a set of endpoints that accept messages containing either document-oriented or procedure-oriented information. The endpoint is defined by a network protocol and a message format, however, the extensible characteristic of WSDL allow the messages and endpoints being described regardless of what message formats or network protocols are being used to communicate. In other words, a WSDL file is an XML document that describes a set of SOAP messages and how the messages are exchanged.

Universal Discovery Description and Integration (UDDI) is the yellow pages of Web services. A UDDI directory entry is an XML file that describes a business and the services it offers. Both can be categorized and have keys so one can find a provider or a service by different ways. Web services are also defined through a document called a Type Model (tModel) that describes their interface. A tModel is simply a WSDL file without the <service> tag; it contains information to generate the different proxy classes or SOAP messages to communicate with the Web service but it does not specify the access point of the service.

VII. Advantages of Web Service Meta Search Engine

We attempt to provide a comprehensive analysis of the potential advantages of metasearch engines over search engines. We will also focus on the comparison of metasearch engine and search engine.

A. *Increased Search Coverage:*

Metasearch engine can search any document that is indexed by at least one of the search engines. Hence, the search coverage of a metasearch engine is the union of those search engines. Metasearch engine with multiple major search engines as components will have larger coverage than any single component search engine. Different search engines often employ different document representation and result ranking techniques, and as a result, they often return different sets of top results for the same user query. Thus, by retrieving from multiple major search engines, a metasearch engine is likely to return more unique high quality results for each user query.

B. *Better Content Quality:*

The quality of the content of a search engine can be measured by the quality of the documents indexed by the search engine. The quality of a document can in turn be measured in a number of ways such as the richness and the reliability of the contents. General-purpose metasearch engines implemented using the large-scale metasearch engine approach has a better chance to retrieve more up-to-date information than major search engines and metasearch engines that are built with major search engines.

C. *Good Potential for Better Retrieval Effectiveness:*

More unique results are likely to be obtained, even among those highly ranked ones, due to the fact that different major search engines have different coverage and different document ranking algorithms. The result-merging component of the metasearch engine can produce better results by taking advantage of the fact that the document collection of major search engines has significant overlaps. This means that many shared documents have the chance to be ranked by different search engines for any given query. If the same document if retrieved by multiple search engines, then the likelihood that the document is relevant to the query increases significantly because there is more evidence to support its relevance. In general, if a document if retrieved by more search engines, the document is more likely to be relevant.

D. *Better Utilization of Resources:*

Metasearch engine use component search engines to perform basic search. This allows them to utilize the storage and computing resources of these search engines.

VIII. Conclusions and Future Work

The goal of this proposal is to shed some light on the reasons that make building metasearch engines, especially large scale metasearch engines, difficult and challenging. Modern search engines providing interfaces that allow external applications to issue Web search queries that are actually processed using their large scale computing infrastructure. This paper proposes a robust Meta search engine, which can communicate to heterogeneous platform using advanced web service techniques. As much progress has been made in advanced metasearch engine technology, several challenges need to be addressed before truly large scale Meta Search Engine can be effectively built and manage.

References

[1] Lee Underwood, “A Brief History of Search Engines”;www.webreference.com/authoring/search_history/.

[2] Ritu Khare, Yuan An and II-Yeol Song on Understanding Deep Web Search Interfaces: A Survey. SIGMOD Record, March 2010 (Vol. 39, No 1).

[3] DogPile metasearch engine. http://www.dogpile.com.

[4] Carol L. Barry. The Identification of User Criteria of Relevance and Document Characteristics: Beyond the Topical Approach to Information Retrieval. PhDthesis, Syracuse, 1993.

[5] Ana B. Benitez, Mandis Beigi, and Shih-Fu Chang. Using relevance feedback in content & based image metasearch. IEEE Internet Computing,2(4):58-69, 1998.

[6] Eric J. Glover, Steva Lawrance, William P. Birmingham and C. Lee Giles on Architecture of the Metasearch Engine that Supports User Information Needs, ACM.

[7] Tieli Sun Wei Zhao and Zhiyan Zhao An Architecture of Pattern-Oriented Distributed Meta-Search Engine ACM.

[8] Amir Hossein Keyhanipoor, Maryam Piroozmand, Behzad Moshiri and Caro Lucas A Multi layer/Multi-Agent Architecture for Meta-Search Engine, AIML 05 Conference, 19-21 December 2005, CICC, Cairo, Egypt.

[9] Jae Hyun Lim, Young-Chan Kim, Hyonwoo Seung, Jun Hwang , Heung-Nam Kim, “Query Expansion for Intelligent Information Retrieval on Internet”, International Conference on Parallel and Distributed Systems, 1997.

[10] Eric J. Glover, Steve Lawrence, William P. Birmingham, C. Lee Giles; “Architecture of a Meta-Search Engine that Supports User Information Needs”, Information and knowledge management CIKM '99, ACM Press,Pages: 210 - 216, 1999.

[11] Sergey Brin and Lawrence Page, “The Anatomy of a Large-Scale Hypertextual Web Search Engine”, World-Wide Web Conference, 1998.

[12] Candadai, “A Dynamic Implementation Framework for SOA-based Applications”, Web Logic Developers Journal: WLDJ, September/October 2004.

[13] UDDI specifications – http://www.uddi.org.

[14] SOAP specifications and RFCs - http://www.w3.org/TR/soap/.

[15] WSDL Specifications and RFCs - http://www.w3.org/TR/wsdl.

[16] T. Clements. Overview of SOAP. Java Developers Forum, http://java.sun.com/developer/technicalArticles/ xml/webservices/.

[17] Ethan Cerami. Web Services Essentials. O'Reilly, February 2002.

[18] K.Srinivas, P.V.S.Srinivas and A.Govardhan. “A Survey on the “Performance Evaluation of Various Meta Search Engines” IJCSI Volume 8, Issue 3,No. 2,May 2011 Pages 359-364.

[19] Nath, R. (2011). An Authorization Mechanism for Access Control of Resources in the Web Services Paradigm. *IJACSA - International Journal of Advanced Computer Science and Applications*, *2*(6), 36-42.