Investigating Web Services on the World Wide Web

Eyhab Al-Masri and Qusay H. Mahmoud
Department of Computing and Information Science
University of Guelph, Guelph, ON, N1G 2W1 Canada
{ealmasri,qmahmoud}@uoguelph.ca

ABSTRACT
Searching for Web service access points is no longer attached to service registries as Web search engines have become a new major source for discovering Web services. In this work, we conduct a thorough analytical investigation on the plurality of Web service interfaces that exist on the Web today. Using our Web Service Crawler Engine (WSCE), we collect metadata service information on retrieved interfaces through accessible UBRs, service portals and search engines. We use this data to determine Web service statistics and distribution based on object sizes, types of technologies employed, and the number of functioning services. This statistical data can be used to help determine the current status of Web services. We determine an intriguing result that 63% of the available Web services on the Web are considered to be active. We further use our findings to provide insights on improving the service retrieval process.

Categories and Subject Descriptors
D.2.12 [**Software Engineering**]: Interoperability - *Distributed objects, Interface definition languages;* H.3.5 [**Information Systems**]: Online Information Services – Commercial services, Data sharing, Web-based services.

General Terms
Management, Design, Measurement, Performance, Verification.

Keywords
UDDI, UDDI Business Registries, Crawler, Web Services, Searching, Crawling, WSCE, WSDL, Interface, Service Portals.

1. INTRODUCTION
At the heart of service-oriented computing is a Web service registry that connects and mediates service providers with clients. Web service registries extend the concept of an application-centric Web by allowing clients or applications to access a wide range of Web services that match specific search criteria. Without publishing Web services through registries, clients will not be able to locate them in an efficient manner, and service providers will have to devote extra effort in advertising these services through other channels.

In recent years, several Web service portals or directories have emerged such as WebServiceList [1], RemoteMethods [2], WSIndex [3], and XMethods.net [4]. However, due to the fact that these Web-based service directories fail to adhere to original Web services' standards such as UDDI, it is likely that they become vulnerable to being unreliable sources for finding or interacting with Web services, and can easily become disconnected from the Web service environments as in the cases of BindingPoint, Woogle, and SalCentral which closed their service portals after many years of exposure. Apart from having service portals, there have been numerous efforts that attempt to improve the discovery of Web services [5,6], however, many of them have failed to address the issue of handling discovery operations across multiple UBRs and other heterogeneous environments. In addition, publishing services across multiple heterogeneous sources (i.e. UDDI, service portals, or search engines) adds another level of complexity with respect to service providers managing and administering them.

Other trends for finding services have also emerged in recent years. Search engines such as Google, Yahoo, AlltheWeb and Baidu have become a new source for finding Web services. However, search engines do not recognize the significance for publishing service information on the Web in such a manner that meets the basic service properties (i.e. binding information, operations, ports, service endpoints, among others).

In addition, search engines generally crawl Web pages from accessible Web sites and since publicly accessible WSDL documents reside on Web servers, they are likely to be fetched by these crawlers. Crawling Web sites for capturing WSDL documents to be indexed by search engines implies that they could be treated as Web pages. However, there are key differences that exist between the structure of Web pages and Web services that make such crawlers unreliable sources for capturing service information. Furthermore, Web-based search engines simply cache or store WSDL documents but they do not provide any business-centric models or adhere to original Web service standards such as the service-oriented computing find-bind-execute paradigm. Nonetheless, search engines may potentially become in the near future very valuable technologies for publishing, searching, and invoking services on the Web.

Based on the above, we conclude that there is a need to establish a targeted Web service crawler engine that can potentially be used for Web services discovery that fits a proper Web services architecture. In this work, we make the following contributions:

- We examine the potential of using service registries for the discovery of Web services versus Web-based search engines, and vice versa.
- We introduce the notion of a targeted Web Service Crawler Engine (WSCE) [7,8,9]. WSCE actively crawls accessible UBRs and search engines to collect business and Web service information. WSCE can also be used for performing service metrics and finding relevant Web services. By continuously crawling existing Web service resources available on the Web, the system is capable of maintaining up-to-date Web

WWW 2008, April 21–25, 2008, Beijing, China.
ACM 978-1-60558-085-2/08/04.

service information, and therefore rendering an effective Web service retrieval.

- We run several experiments on a large dataset consisting of the plurality of Web services that can be accessed on the Web today.
- We collect and analyze results and present various statistics including how many services are accessible; how many are functioning; object size distribution; and technology trends in employing Web services.

The rest of this paper is organized as follows: Section 2 describes some of the related work. In Section 3 the methodology of our research is discussed. Section 4 describes the architecture of WSCE. Section 5 discusses some of the main requirements and challenges for crawling Web services. Results and evaluation from WSCE are discussed in Section 6. Section 7 describes some of the challenges in the discovery of services. Finally conclusion and future work are discussed in Section 8.

2. RELATED WORK

Discovery of Web services is of an immense interest and is a fundamental area of research in ubiquitous computing. Many researchers have focused on discovering Web services through a centralized UDDI registry [10,11,12]. Although centralized registries can provide effective methods for the discovery of Web services, they suffer from problems associated with having centralized systems such as a single point of failure, and bottlenecks. In addition, other issues relating to the scalability of data replication, providing notifications to all subscribers when performing any system upgrades, and handling versioning of services from the same provider have driven researchers to find other alternatives.

Other approaches focused on having multiple public/private registries grouped into registry federations [6,13] such as METEOR-S for enhancing the discovery process. METEOR-S [6] provides a discovery mechanism for publishing Web services over a federated registry sources but, similar to the centralized registry environment, it does not provide any means for advanced search techniques which are essential for locating appropriate business applications. In addition, having a federated registry environment can potentially provide inconsistent policies to be employed which will significantly have an impact on the practicability of conducting inquiries across the federated environment and can at the same time significantly affect the productiveness of discovering Web services in a real-time manner across multiple registries.

Other approaches focused on the use of text document matching [14,15] and mainly depend on analyzing the frequency of terms. Other research attempts focused on schema matching [16,17] which try to understand the meanings of the schemas and suggest any related matches. Other research studies examined the potential of using supervised classification and unsupervised clustering for Web services [18], artificial neural networks [19], or unsupervised matching at the operation level [20].

Some other approaches focused on the peer-to-peer framework architecture for service discovery and ranking [21], providing a conceptual model based on Web service reputation [22], and providing keyword-based search engine for querying Web services [23]. However, these approaches provide a very limited set of search capabilities (i.e. search by business name, business location, etc.) that would make it impractical for clients to perform proper service queries tailored to their needs.

Although keyword matching methods (i.e. broad, phrase, exact, and negative) may partially support the discovery of Web services, they do not provide clients with efficient ways for articulating proper service queries (i.e. consider input/output values of service operations). In addition, the lack of having the appropriate methods and tools to search for Web services contributes significantly to the scarcity in determining the current status of Web services.

Very little research is conducted on investigating Web services on the Web. In [28], the authors provide an exploratory study on Web services on the Web. The study provides some details and statistics from Web services collected throughout the Web via Google API such as operation analysis, size analysis, words distribution, and function diversity analysis. However, the study does not provide a complete view of Web services on the Web and focuses only on a single search engine. This may provide only a cross section of what is available on the Web today and therefore may provide inaccurate or misleading conclusions. In addition, the number of services claimed constitutes only 67% of our dataset.

In another effort, authors in [31] describe a programmatic approach to Web service brokerage and provide some statistics on the available services in their Merobase repository. However, the study does not provide any statistical analysis on the number of functioning services or how they were collected. It is imperative to determine if the number of services collected is from a single service resource on the Web or from multiple environments. In addition, using WSCE, we have been able to collect a larger selection of Web services than Merobase which accounts only 60% of our dataset. In addition, the study does not provide any details on how many services were functioning, how they were collected, or any historical distribution.

Recently, some Web-based directories such as RemoteMethods or WebServiceList focused on providing simple service portals based on a keyword search paradigm of Web service descriptions. However, due to the fact that organizations can develop custom taxonomies created for specialized use within a business which serves as a tagging mechanism for UDDI service entries with critical metadata, it becomes apparent that simple keyword search methods are inefficient. For example, this metadata is not well organized or the keyword search method does not capture the underlying semantics of this metadata, Web service data will not be easily discovered and therefore, results will not yield meaningful information. In fact, if Web service-related data cannot be understood, its functionality is considered non-existing or misleading. In addition, a user who is not able to determine the context of a given Web service (i.e. the host location, how it is supported, etc.), then the user will not be able to effectively interact with the Web service.

3. METHODOLOGY

The procedure of our research consists of: (1) building a database for the majority of Web service resources that are accessible including UBRs, service directories/portals, and search engines indices; (2) building the necessary tools to automatically crawl service resources and collecting Web service information including metadata and WSDL documents; (3) recursively parse the underlying Web service interfaces; (4) verify and validate the

number of accessible Web services; and (5) analyze retrieved Web services by obtaining service metadata, WSDL documents, supported protocols, serviceability, types of technologies employed, among others.

Although there are several sources that are widely used in the research community for accessible Web sites such as Alexa Inc. [29] and Stanford WebBase project [30], there is very little or no known source that provides comprehensive list of sources for accessible Web services. Prior to building the necessary tools to crawl service resources using the Web Service Crawler Engine (WSCE), it was necessary to first build a set of available resources that provide a list of Web services that can later be used as the input to our experiments. By combining lists of Web services from these available Web service resources, we are able to achieve a much larger dataset and aggregate a much larger number of Web services. Unfortunately, studies such as [28] have primarily focused on a particular Web service resource, a cross section of services available over the Web today, or do not clearly state how services were collected [31] which may not provide a comprehensive statistical analysis on all possible resources and hence may provide inaccurate or misleading results or conclusions.

4. Web Service Crawler Engine (WSCE)

Applying Web crawling techniques to Web service definitions or WSDL files, and business registries or UBRs may not be efficient, and the outcome of our research was an enhanced crawler targeted for Web services. The crawler should be able to handle WSDL files, and UBR information concurrently. In addition to that, the crawler should be able to collect this information from multiple registries and storing them into a centralized repository, the Web Service Storage (WSS) [9]. WSS serves as a central repository where data and templates discovered by the Web Service Crawler Engine (WSCE) are stored. WSS represents a collection or catalogue of all business entries and related Web services.

WSS plays a major role in the discovery of Web services in many ways: first: it enables for the identification of Web services through service descriptions and origination, processing specification, device orientation, binding instructions, and registry APIs; second: it allows for the query and location of Web services through classification; third: provides means for the service life-cycle tracking; fourth: provides dynamic service invocation and binding; fifth: provides means for the advanced search queries; sixth: enables the provisioning of Web services by capturing multiple data types, and seventh: provide means for the advanced business notification for Web services. The WSS also takes advantage of some of the existing mechanisms that utilize context-aware information using artificial neural networks [19] for enhancing the discovery of Web services.

Our approach in implementing this conceptual discovery model shown on Figure 1 is a process-per-service design in which WSCE runs each Web service crawl as a process that is managed and handled by the WSCE's Event and Load Manager (ELM). The crawling process starts with dispensing Web services into the WsToCrawl queue. WSCE's SeedWs list contains hundreds or thousands of Web services with their corresponding access points.

WSCE begins with a collection of Web services and loops through taking a Web service from WsToCrawl queue. WSCE then starts analyzing Web service information located within the

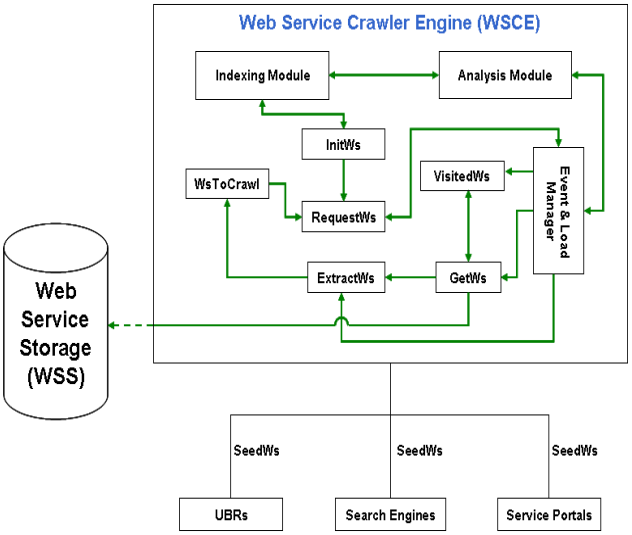

Figure 1. Web Service Crawler Engine (WSCE)

registry, tModels, and any associated WSDL information through the Analysis Module (AM). WSCE stores this information into WSS after processing it through the Indexing Module (IM). IM is primarily responsible for building data structures over textual information contained within WSDL interfaces or UDDI objects (i.e. businessEntity, businessService, bindingTemplate, tModels, among others). After completion, WSCE adds an entry of the Web service (using serviceKey) into VisitedWs queue. More details on the complete architecture and components of WSCE can be found in [7,9].

4.1 Web Service Resources

Finding information about Web services is not strictly tied to UBRs. There are other standards that support the description, discovery of businesses, organizations, service providers, and their Web services which they make available, while interfaces that contain technical details are used for allowing the proper access to those services. For example, WSDL describes message operations, network protocols, and access points to addresses used by Web services; XML Schemas describe the grammatical XML structure sent and received by Web services; WS-Policy describes general features, requirements, and capabilities of Web services; UDDI business registries describe a more business-centric model of Web services; WSDL-Semantics (WSDL-S) uses semantic annotations that defines the meaning of inputs, outputs, preconditions, and effects of operations described by a Web service interface. The following sections briefly describe the plurality of the possible resources for collecting Web services on the Web.

4.1.1 UDDI Business Registries (UBRs)

UBRs are used for publishing and discovering Web services into registries. There are several key UBRs that currently exist and were used for this method including: Microsoft, XMethods, SAP, National Biological Information Infrastructure (NBII), among others.

4.1.2 Web-based

Web-based crawling involves using an existing search engine API to discover WSDL files across the Web such as Google and Yahoo search APIs. Using this method, a crawler engine can

continuously parse search results from an existing search engine when looking for Web services throughout their indices. This involves the use of search engine specific features to collect Web service information. For example, Google Search API [26] provides a way to search for files with any extension such as WSDL, DISCO, or WSIL. There were several key search engines indices that were used for crawling these types of service resource including: Google, Yahoo, AlltheWeb, and Baidu.

4.1.3 File Sharing

File sharing tools such as Kazaa and Emule provide search capability by file types. Similar to search engines, file sharing tools may provide a way to collect Web services. However, unlike search engines, peer-to-peer file sharing platforms provide variable network performances, the amount of information being shared is partial, and availability of original sources could not be guaranteed at all times which prompted us to exclude this method from crawling.

4.1.4 Service Portals or Directories

One possible method for collecting Web services is through Web-based service directories or portals such as Woogle [20], WebServiceList [1], RemoteMethods [2], and others. Capturing Web services from service portals requires public access to their repositories or building custom crawlers designed to capture Web service data from each portal independently which prompted to exclude this method from WSCE. In addition, the majority of Web services listed within these directories were either indexed by search engines or listed in existing UBRs. Unfortunately, many of these Web-based service portals do not adhere to the Web service standards, and therefore it becomes impractical to use them for crawling Web services.

4.2 Dataset

Web service information is not strictly tied to service interfaces. UDDI provides a more business-centric structure for publishing service and business information while other defined Web service resources only provide 'links' or access points to service interfaces or WSDL documents. Therefore, a Web service is not simply a WSDL document, but rather other metadata information that were carefully considered when creating service registry specifications such as UDDI or ebXML. However, for the purpose of this study, only WSDL documents are considered since the majority of services were obtained through search engines.

We developed a crawling strategy that would accommodate each service resource in which we obtained a list of 7,591 possible Web service interfaces. However, it was necessary to refine this crawling strategy due to inconsistent search results obtained from these resources or their inability to validate the integrity of service references stored in their databases. Therefore, it was necessary to apply several intelligent crawling techniques to filter out inaccurate services. Through the refined technique, we were able to obtain 5,077 unique WSDL references. Additional filtering techniques were necessary to determine the number of valid service interfaces for which we have built three types of crawler tools: (1) VerifyWS, (2) ValidateWS, and (3) MetaCollector.

4.3 Tools

The crawling tools consist of a verifier, validator, and metadata collector. A Web service is passed to the WSCE crawler tools after a resource is examined. Crawlers are used to build the backend index for search engines by following links from one page to another. However, Web service crawling is relatively distinctive from Web page crawling (later discussed in Section 7).

4.3.1 VerifyWS

After crawling Web services of a particular resource as shown in Figure 1, WSCE uses the VerifyWS (via InitWs) to determine whether a WSDL reference is an active URL or not. Once a service reference is verified, the crawler compiles a hierarchical list of WSDL references that are considered to have active URLs and passes it to the ValidateWS.

4.3.2 ValidateWS

Referencing a WSDL document in a UBR or having a link appearing in a search engine result does not necessarily imply that the reference reflects an actual or real service interface. Therefore, it is often necessary to continue filtering the crawling process to validate the content of a WSDL document and determine whether it is a real service interface or not. ValidateWS (via RequestWS) validates a service interface URL by parsing the content and performs a series of tests to determine the grammatical structure of a service interface. ValidateWS is the first step that enables WSCE to begin indexing and analyzing service information (Figure 1) and/or any possible measurements that could apply such as QoS metrics [27]. Once a service interface is validated, the crawler compiles a hierarchical list of WSDL references that are considered to be functional services and begins collecting any associated metadata.

4.3.3 MetaCollector

Once the validation of a WSDL document is complete, the MetaCollector (via GetWs) begins capturing information contained within the interface (i.e. operation names, message names, among others) and any additional information provided by the resource. If a resource is determined to be a UBR, the crawler through the MetaCollector intelligently begins capturing service and business information from the registry. If the resource is determined to be a search engine results, it captures all possible information that a search engine can retrieve (i.e. summary snippet, cache size, title, among others).

5. WSCE DESIGN CHALLENGES

Based on our experience with implementing WSCE and given the proliferation and change rate of Web services, service crawlers need to take into consideration many factors and challenges that are discussed in this section.

5.1 Types of Web Services to Download

At many instances, crawlers cannot retrieve or download all Web services that exist on the Web. At some instances, service providers may require authentication for clients to browse through their service registries as in the case of the US Environmental Protection Agency UDDI Registry in which access to the registry is protected and controlled based on roles. However, this is similar to situations in which Web crawlers attempt to crawl secure content on the Web in which they must authenticate in order to download Web pages within secured locations. Given this fact, it is important for a Web services' crawler to carefully select and identify the sources and types of Web services and to retrieve those that are considered important first so that the fraction of Web services retrieved over the Web is more meaningful.

5.2 Crawler Update

At many instances, service providers may update their Web services including description and WSDL information. Crawlers need to be able to update or revisit Web services periodically in order to determine changes that may have taken place and update the collection of downloaded information. In order for a crawler to achieve a particular refresh status, it needs to determine a collection of Web services that must be revisited while skipping those that are considered less frequently updated. To illustrate how the update rate works, a crawler will take into consideration those Web services that are often updated in which case it will revisit them more frequently.

5.3 Scalability

As the number of Web services increases, having one crawler engine or WSCE may not be efficient. In this case, it would be desirable to provide a mechanism for distributing crawlers across multiple machines. This process is often necessary to download a large number of Web services across one or more UBRs, service portals, or search engines. Therefore, there is a need for a mechanism that organizes the coordination between these crawlers so that they do not download the same information multiple times.

5.4 Load Minimization

At many instances, a crawler may need to collect Web services from existing UBRs or Web servers. In this case, the crawler will consume resources that belong to other organizations who may complain or block access by the crawler. To avoid such cases, the crawler has to be intelligent enough to minimize the load on network and other organization resources.

6. RESULTS

In this section we present results and statistics about collected Web services during one of the routine daily crawls of WSCE described in Section 4. The dataset is distributed among 4 machines, running experiments simultaneously. Each machine has a 3.1 GHz Pentium IV running on Windows 2000. Table 1 provides details of the dataset with respect to the resources used to collect Web services.

Table 1. Dataset for WSCE

	UBRs	Search Engines
Crawled Services	1405	3672
Execution Time (sec)	3640	4300
Total Crawled Services	5077 services	

6.1 Service Growth and Distribution

The total number of Web services crawled in one of WSCE's latest crawls is 5077 services, the majority of which can be found through search engines. Results from an earlier crawl by WSCE in October 2006 show that the number of services collected through UBRs significantly exceeded that of search engines.

Search engines have grown significantly in the last few months providing a much larger number of Web services compared to earlier crawls by WSCE. In fact, comparing the number of services from October 2006 to the time this paper is written (October 2007), the number of Web services collected through WSCE in October 2007 is approximately 10.4 times larger (with respect to UBRs) while the growth rate of Web services through UBRs was only ridiculously insignificant when compared to search engines. Table 2 compares the number of Web services from two different WSCE crawls.

Table 2. Historical service distribution (2006 and 2007)

	October '06	October '07	Growth Rate
UBRs	1248	1405	+ 12.6 %
Search Engines	951	3672	+ 286 %
Total	2199	5077	+ 131 %

Table 2 shows the growth rates for services for one year. During this period, a significant number of services captured using WSCE through search engines. During October 2006, search engines comprised of approximately 43% of the available services during a WSCE crawl. However, the growth rate for search engines with respect to indexing services has grown significantly in which 72% of the services during a recent WSCE crawl are captured through search engines (Tables 1 and 2). In addition, the growth rate of search engines in terms of finding WSDL references has grown by 286% while those of UBRs grew by only 12.6%. Furthermore, the 131% overall increase in the number of services over this year period demonstrates that Web services are becoming more popular. On the contrary, this significant increase reflects the slow growth of UDDI since its formation.

Nonetheless, supporting UDDI and enabling organizations to self-operate and manage their own service registries is evident as new operating systems, applications, and APIs are equipped within built in functionalities or tools for allowing businesses or organizations to create their own internal service registries for intranet and extranet use such as Enterprise UDDI Services in Microsoft Windows 2003 Server, IBM WebSphere Application Server, Systinet Business Service Registry, jUDDI, to name a few. However, many service vendors such as Salesforce and others are not using UDDIs due to existing limitations which gradually led to the creation of Web-based service portals (i.e. Salesforce AppExchange/Force.com, XMethods.net, and ESynaps.com) and potentially using Web-based search engines (i.e. Google, Baidu, Yahoo) for Web service discovery.

Such slow growth in the number of UBRs and the fast growth of search engines in providing references to available services may provide the potential of having search engines as the next major player to discovering Web services on the Web. Whether the number of service registries, service portals, or service interfaces increases, the fact remains that discovering and selecting Web services through a standard, universal access point facilitated by a targeted Web services' crawler is inherent and inevitable.

6.2 Verifying WSDL Interfaces

After collecting Web service interfaces, it is important to determine the number of active URLs. Therefore, for every Web service collected, a verification test is applied using the VerifyWS component of WSCE. Web services that fail the verification test are excluded from additional steps within WSCE (i.e. indexing module, analysis module, etc.). Table 3 presents a breakdown of the result from the VerifyWS crawling tool.

Table 3. VerifyWS crawling test results

	UBRs	Search Engines	Total
Active URL	661	3372	4033
Inactive URL	744	300	1044
Total	1405	3672	5077

Of the 5077 Web service interfaces collected, 79% of them are

considered to have active links. Figures 2 and 3 present active URLs WSDL distribution for UBRs and search engines, respectively.

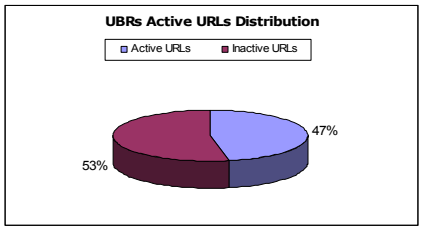

Figure 2. WSCE verification test results for UBRs

As can be seen from Figure 2, the number of inactive URLs exceeds that of active ones. Based on the data collected through UBRs, many of the services published through them were mostly for testing purposes, and hence may contain inaccurate information. At many instances, service metadata information collected through UBRs contained irrelevant service descriptions or did not have any relevant details. Unfortunately, due to the fact that many of the information contained within UBRs is not accurate, it consumed more resources when performing tests and took longer to execute which is reflected by execution time for UBRs in Table 1, and can be interpreted by the high percentage of inactive links.

Although the number of Web services crawled from each approach varies, Web-based search engine results appear to have a much higher percentage of active links for WSDL documents than UBRs due to a variety of reasons most importantly that search engines have an update interval that checks for any outdated links and hence exclude them from returned search results. Unfortunately, registration for accessible UBRs is voluntary and therefore many access points may be broken or outdated since no such mechanism exists that can determine the validity of access points at the time of registration or continuously checks for any outdated links.

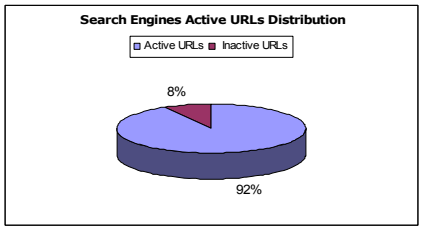

Figure 3. WSCE verification test results for search engines

As can be seen in Figure 3, information collected through search engines does not imply that links provided in their search results can always be assumed to have active links. In fact, 8% of returned results contained outdated or inactive URLs. The majority of the 300 inactive URLs were mostly captured from Yahoo's and Baidu's search engine results. Google and AlltheWeb had very few inactive URLs in their search result which indicates that their crawlers visit Web sites and crawls pages more often.

6.3 Validating WSDL Interfaces

After excluding inactive WSDL files based on our verification test, we apply a validation test that determines the number of interfaces that represent real Web service implementations. For example, having an active URL for a WSDL document does not imply that the URL reflects an actual Web service interface and it could be an XML or HTML file with the same file extension. This step helps amplify WSDL documents in order to remove any files with "WSDL" extension but do not conform an actual WSDL file schema. In addition, the validation step (which uses ValidateWS component) guarantees to remove any redundant Web service interfaces fetched during the crawling process. Furthermore, ValidateWS helps determine the number of Web services per host (i.e. domain name), and the number of Web services fetched from a particular domain type (i.e. commercial, network, organization, among others). The results from the ValidateWS crawling test are shown in Table 4.

Table 4. ValidateWS crawling test results

	UBRs	Search Engines	Total
Valid	242	2942	3184
Invalid	419	430	849
Total	661	3372	4033

Of the 4033 Web services that passed our VerifyWS step, 79% of them are considered valid service interfaces and conform to a WSDL schema. Figures 4 and 5 present valid WSDL distribution for UBRs and search engines. However, as discussed in earlier, search engines were not designed for collecting information about Web services, and therefore, they do not offer a complete solution for discovering Web services. Search engines only provide pointers to access points (or WSDL documents) which is one of the many components in the overall Web services architecture.

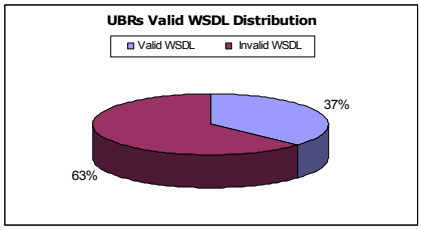

Figure 4. WSCE validation test results for UBRs

Results from Figure 4 coincide with those in Figure 2. Although the majority of Web services contained in UBRs during the verification test were inactive URLs, this fact is also true when validating active URLs. Only 37% of the 661 Web services that could be verified can be validated as valid WSDL documents. This indicates that only 17% of the Web services initially collected through UBRs after the first crawling process can be considered actual Web service implementations.

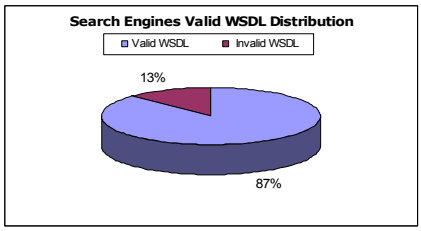

Figure 5. WSCE validation test results for search engines

As can be seen in Figure 5, using search engines, 87% of the verified Web services were successfully validated. This indicates that 80% of the Web services initially collected through accessible search engines after the first crawling process can be considered functional. Although this percentage represents an

Table 5. Breakdown of HTTP status codes (VerifyWS)

Code	Description	UBRs		Search Engines	
		# Web Services	Percent	# Web Services	Percent
200	OK	661	47.05	3372	91.83
301	Moved Permanently	0	0.00	1	0.03
400	Bad Request	3	0.21	3	0.08
401	Unauthorized	0	0.00	1	0.03
403	Forbidden	8	0.57	4	0.11
404	Not Found	231	16.44	40	1.09
405	Method Not Allowed	11	0.78	0	0.00
406	Not Acceptable	4	0.28	7	0.19
411	Length Required	8	0.57	0	0.00
500	Internal Server Error	38	2.71	15	0.41
502	Bad Gateway	12	0.85	2	0.05
503	Service Unavailable	1	0.07	2	0.05
Others	Remote Server Error	254	18.08	219	5.97
Others	Invalid URI	174	12.38	6	0.16
Total		1405	100.0	3672	100.0

error value of 20% (or 730 services), it is nonetheless high. By combining these findings, we determine that the success rate of finding a real or functional service implementation is 63% which implies that searching for Web services can become time consuming. Furthermore, from these findings, we note that additional amplification steps will provide more consistent search results but they do not involve any measures as to the relevancy of finding a Web service of interest.

Due to the fact that the majority of service interfaces collected contained little or no documentation of what they offer, it becomes challenging and time consuming to discover relevant Web services. Although statistics from our experiments show that search engines may provide a much larger selection for finding Web services, they may become vulnerable to returning irrelevant search results mainly due to the fact that information retrieval techniques applied to Web pages could not simply be used for Web services and may fail to retrieve relevant results. When looking for appropriate Web services, clients look for those that meet their requirements particularly the overall functionality and Quality of Service (QoS) [27].

6.4 HTTP Status Distribution

We believe that investigating HTTP status distribution for collected Web services in our dataset can provide an overview of the current status of Web services. HTTP status distribution can also provide many details such as how many services need authentication prior to consuming them, fail to execute properly, could not be found, or have network related issues (i.e. down time, server unavailability, among others). Table 5 provides a breakdown of the HTTP status codes for WSDL documents after the WSCE verification test (using VerifyWS crawling tool).

Results from Table 5 show that the majority of errors for both UBRs and search engines occur in the "Remote Server Error" which could be due to a variety of reasons such as the remote host was down at the time this test has taken place or some other network failure. In both approaches, WSCE was able to verify the majority of Web services as in the case of UBRs in which 661 Web services were verified (which accounts for 47% of crawled Web services) and in the case of search engines in which 3372 Web services were successfully verified (which accounts for 92% of crawled Web services). However, WSCE verification test shows that 16.44% of Web services in UBRs contain an inactive or broken links while this condition accounts for only 1.09% of Web services crawled through search engines. This is due to the fact that search engines have a higher refresh rate that would eventually exclude broken links while UBRs do not enforce such mechanism and therefore UBRs may contain a much higher number of outdated or broken links than search engines.

6.5 WSDL Size Distribution

We believe that the WSDL file distribution could provide an overview of the current status of Web services on the Web, their magnitude, level of complexity, and file size comparison to Web pages. To achieve this task, an additional WSDL-content test was performed to determine the average size of WSDL documents that were successfully crawled and Figure 6 presents the results from this test. Of the 661 WSDL documents that were successfully downloaded by the WSCE from UBRs, 83% were between 1K and 64K bytes in size while of the 3372 WSDL documents that were successfully downloaded by the WSCE from search engines, 91% were between 1K and 64K bytes in size.

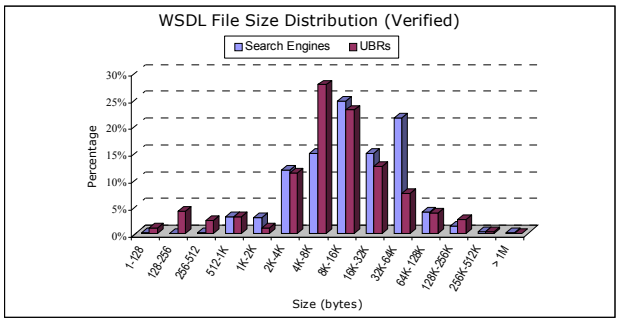

Figure 6. WSDL file size distribution (verification)

Figure 6 presents a histogram of the WSDL document size distribution. In this figure, WSDL documents were distributed across fourteen bins labeled with increasing the document size exponentially in which a WSDL document of size m is likely to be placed in a bin that is not greater than the value. Figure 7 presents a histogram of the distribution of Web services that passed the validation test.

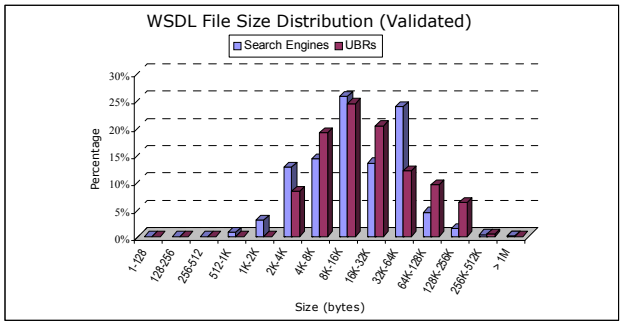

Figure 7. WSDL file size distribution (validation)

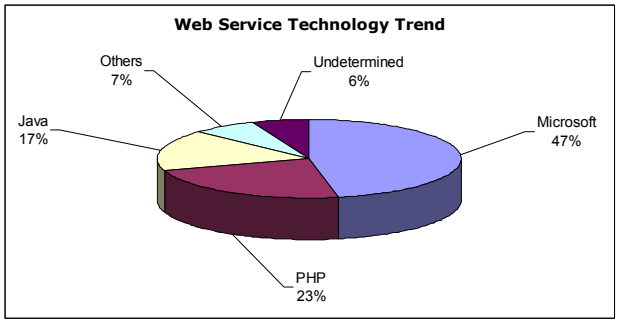

Figure 8. Types of technologies used for employing services

Figure 7 presents a histogram showing the document size distribution of WSDL documents that were validated and complied to a WSDL specification (i.e. contained the necessary tags) in which 84% were between 1K and 64K bytes in size in the case of UBRs and 93% in the case of search engines.

6.6 Development Technology Trends

In this study, we have come to another intriguing finding that relates to the types of technologies used for building Web services. Although Web service interfaces are meant to block or hide access to application code (service endpoint) and hence communication between services is language independent, we can determine the technology used for building application code. All of the technologies used to build Web services provide ways to automatically generate WSDL documents and therefore have their own propriety format on how they are created. To illustrate how we can determine the type of technology used, consider for example how Microsoft .NET generates WSDL documents by simply appending "?WSDL" to a Web service URL. Some other technologies are more complex to determine such as Java (i.e. service endpoint could be a folder name). However, by examining the location of WSDL documents on the Web (i.e. URLs) and parsing service endpoint, we can determine to an extent the type of technology used to generate them.

Based on our data, the majority of Web services collected were implemented using Microsoft .NET technology. In fact, 47% of them appear to be created using ASP.NET followed by PHP with 23%, and Java with 17%. Other technologies such as ColdFusion, Common Gateway Interface (CGI) among others constituted 7% while the remaining 6% was undetermined. This is due to the fact that at some instances WSDL files may contain invalid service endpoints which makes the task of determining the type of technology applied unknown. However, results from our study provide to some extent an overall view of the most preferred technologies used for building Web services. Figure 8 shows a breakdown of the types of technologies used when examining WSDL documents through the analyzer module.

7. DISCUSSION

7.1 Challenges in Web Services Discovery

Business registries provide the foundation for the cataloging and classification of Web services and other additional components. UDDI Business Registry (UBR) serves as the central service directory for the publishing of technical information about Web services [24]. The current design of the UDDI allows for simplified search capabilities and provides a minimal control for trading partners to publish related business data and categorization for their Web Service advertisements.

Although the UDDI provides ways for locating businesses and how to interface with them electronically, it is limited to a single search criterion. The simplified search techniques offered by the UDDI will make it impractical to assume that it can be very useful for Web services' discovery or composition. In addition, a client does not have to endlessly search UDDI registries for finding an appropriate business application. As Web services proliferate and UBRs becomes filled with hundreds o thousands of Web services, limited search capabilities will likely yield meaningful search results which makes the task of performing search queries across one or multiple UBRs very time consuming, and less productive.

Apart from the problems regarding limited search methods offered by UDDI, there are other major limitations and shortcomings with the existing UDDI. Some of these limitations include: (1) UDDI was intended to be used only for Web services discovery; (2) UDDI registration is voluntary, and therefore, it can easily become passive; (3) UDDI does not provide any guarantee to the validity and quality of information it contains; (4) the disconnection between UDDI and the current Web; (5) UDDI is incapable of providing Quality of Service (QoS) measurements for registered Web services, which can provide helpful information to clients when choosing appropriate Web services, (6) UDDI does not clearly define how service providers can advertise pricing models; and (7) UDDI does not maintain nor provide any Web service life-cycle management.

WSDL documents are no longer a scarce resource as there are thousands of Web services disseminated throughout the Web and not necessarily through UBRs. Due to the fact that Web services are syntactically described through Web Services Description Language (or WSDL) documents which reside on Web servers, such documents can potentially be indexed by Web-based crawlers. This allows search engines to enable users to perform search queries for discovering Web services disseminated throughout the current Web.

However, Web-based search engines crawl such document types on the assumption that they contain textual information that can be indexed or treat them in the same manner as Web pages which makes search engines incapable of indexing Web services. Unfortunately, a considerable amount of WSDL files crawled over the Web as discussed in Section 4 did not contain descriptions of what these Web services have to offer and a considerable amount of the crawled Web services contained outdated, passive, or incomplete information.

7.2 Information Retrieval and Web Services

Collecting Web services data is not the key element that leads to an effective Web services' discovery, but how it is stored. The fact that Web services data is spread all over existing search engines databases, accessible UBRs, or file sharing platforms does not mean that clients are able to find these Web services without difficulties. However, making this Web services data available from a standard, universal access point that is capable of aggregating this data from various sources and providing clients to execute search queries tailored to their requirements via a search engine facilitated by a Web service crawler engine or WSCE is a key element to enhancing Web services discovery and accelerating the adoption of Web services.

Crawling for Web services is very complex and requires special attention particularly looking at the current Web crawler designs. When designing WSCE, it became apparent that many of the existing information retrieval models that serve as basis for Web crawlers may not be very suitable when it comes to Web services due to key differences between Web services and Web pages including:

- Web pages often contain long textual information while Web services have very brief textual descriptions of what they offer or little documentation on how it can be invoked. This lack of textual information makes keyword-based searches vulnerable to returning irrelevant search results and therefore become very primitive means for effectively discovering Web services.
- Web pages primarily contain plain text which allows search engines to take advantages of information retrieval methods such as finding document and term frequencies. However, Web services structure is much more complex than that of Web pages and only a small portion of plain text is often provided either on UBRs or service interfaces which makes the dependency on information basic retrieval techniques very unreliable since they were intended for this type of complex structures.
- Web pages are built using HTML which has a predefined or known set of tags. However, Web service definitions are much more abstract. Web service interface information such as message names, operation and parameter names within Web services can vary significantly which makes the finding of any trends, relationships, or patterns within them very difficult and requires excessive domain knowledge in XML schemas and namespaces.

Applying Web crawling techniques to Web service definitions or WSDL files, and business registries or UBRs may not be efficient, and the outcome of our research was an enhanced crawler targeted for Web services. The crawler should be able to handle WSDL files, and UBR information concurrently. In addition to that, the crawler should be able to collect this information from multiple registries and storing them into a centralized repository, the Web Service Storage (WSS) [9]. WSS serves as a central repository where data and templates discovered by the WSCE are stored. Table 6 outlines a comparison between UBRs and Web-based search engines used for the discovery of Web services.

The ability to explore service registries and Web-based search engines for finding appropriate Web services is becoming a challenge. Although the UDDI has been approved as a standard for Web service discovery, the lack of autonomous control acts considerably as a deterrent for the widespread its deployment [25]. In addition, Web-based search engines were not designed to handle the Web services syntactic structure; therefore they only provide an ad-hoc solution for matching keywords that appear within WSDL documents.

Table 6. Comparison between UBRs and search engines

Features	UBRs	Search Engines
Contains business information?	Yes	No
Uses tModels?	Yes	No
Is publishing (listing) voluntary?	Yes	Yes
Any service-like structure?	Yes	No
Stores WSDL Documents	No	Yes
Any update interval?	No	Yes
Any support for range-based searching?	No	No
Any support for caching?	No	Possibly
Search Capabilities	Limited	Keyword matching
Any Web service subscription/business model?	Yes	No
Can handle versioning?	No	No
Validates, governs, or secures Web services?	No	No
Any support for Web service specific measurements?	No	No

8. CONCLUSION

On the dawn of service-oriented computing, finding relevant Web services was mainly done by scanning through services registries (i.e. UDDI Business Registries or UBRs). Automated Web service search engines were not necessary when Web services were counted by the hundreds. However, the number of service registries is gradually increasing and Web service access points (i.e. WSDLs) are no longer a scarce resource as there are thousands of Web services disseminated throughout the Web.

Our experiments show building a crawler and a centralized repository for Web services is inevitable. In this work, we have used our Web Service Crawler Engine (WSCE), a crawler that is capable of capturing service information from various accessible resources over the Web, to help us in conducting our investigation of Web services on the Web.

In our study, we investigated the distribution of certain elements and characteristics of the available Web services on the Web. Distribution based on valid WSDL interfaces, file sizes, HTTP status, and technology trends are found. Results provide an overall view on the current status of Web services. An intriguing result is that fact those search engines have become a new major source for searching for Web services and that they constitute 72% of Web services available on the Web. Such service statistics may likely drive search engines to examine the potential of interoperability with service registries or apply features that can turn them into effective tools used for discovering services on the Web.

Although UDDI and search engines provide two distinctive approaches for finding Web services, it is unclear whether they will likely merge or coexist. Based on our findings, search engines have become a new major source for searching for Web services. Yet, they are vulnerable to returning irrelevant results and only provide access points to WSDL documents while UDDI business registries provide a more business-centric model that can

be used as the first step towards an application-centric Web. In addition, results show that collecting Web service information is beyond simple crawling and information retrieval techniques and therefore they may not be applicable at the Web services level. Searching for Web services based on QoS parameters, schema properties, service reputation, trust, and semantic matching will considerably increase the relevancy of finding and selecting appropriate Web services.

Interoperability among existing technologies used for discovering Web services would complement the strengths of each other, although the ability to administer, manage, and search for Web services in a uniform fashion across heterogeneous environments remains an obstacle as services proliferate. Future work includes extending the notion of a targeted Web services crawler engine to continuously perform Quality of Web Service (QWS) metrics on collected Web services, enable clients to selectively control the discovery process, and rank relevant Web services.

9. ACKNOWLEDGMENTS

We would like to thank Mohamed Al-Masri for his valuable time and effort in collecting and analyzing data for this work. This research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant No. 045635.

10. REFERENCES

[1] Web Service List, http://www.webservicelist.com, Accessed February 2008.

[2] RemoteMethods: Home of Web Services, http://www.remotemethods.com, Accessed February 2008.

[3] Web Services Directory (WSIndex), http://www.wsindex.org, Accessed February 2008.

[4] XMethods, http://www.xmethods.net, Accessed February 2008.

[5] E. Maximilien and M. Singh. "Conceptual model of Web service reputation," ACM SIGMOD Record, 31(4), 2002.

[6] K. Sivashanmugam, K. Verma, and A. Sheth, "Discovery of web services in a federated registry environment," ICWS, pp. 270-278, 2004.

[7] E. Al-Masri, and Q.H., Mahmoud, "A framework for efficient discovery of web services across heterogeneous registries," IEEE Consumer Communication and Networking Conference (CCNC), pp. 415-419, 2007.

[8] E. Al-Masri, and Q. H. Mahmoud, "Crawling multiple UDDI business registries", 16th WWW Conf., pp. 1255-1256, 2007.

[9] E. Al-Masri, and Q. H. Mahmoud, "WSCE: A crawler engine for large-scale discovery of web services," ICWS pp. 1104-1111, 2007.

[10] U. Thaden, W. Siberski, and W. Nejdl, "A semantic web Based Peer-to-Peer Service Registry Network," Technical Report, Learning Lab Lower Saxony, 2003.

[11] M. Paolucci, T. Kawamura, T. Payne, and K. Sycara, "Semantic matching of web services capabilities," ISWC, pp. 333-347, 2002.

[12] M. Paolucci, T. Kawamura, T. Payne, and K. Sycara, "Importing the semantic web in UDDI," International Workshop on Web Services, E-Business, and the Semantic Web, pp. 225-236, 2002.

[13] C. Zhou, L. Chia, B. Silverajan, and B. Lee, "UX- an architecture providing QoS-aware and federated support for UDDI," ICWS, pp. 171-176, 2003.

[14] L. Larkey, "Automatic essay grading using text classification techniques," ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 90-95, 1998.

[15] Y. Yang, and J. Pedersen, "A comparative study on feature selection in text categorization," Fourteenth International Conference on Machine Learning, pp. 412-420, 1997.

[16] S. Melnik, H. Garcia-Molina, and E. Rahm, "Similarity flooding: a versatile graph matching algorithm," 18th International Conference on Data Eng, pp. 117-128, 2002.

[17] E. Rahm and P. Bernstein, "A survey on approaches to automatic schema matching," The International Journal on Very Large Databases 10(4), pp. 334-350, 2001.

[18] A. Heß and N. Kushmerick, "Learning to attach semantic metadata to web services," ISWC, pp. 258-273, 2003.

[19] E. Al-Masri, and Q.H. Mahmoud, "A Context-Aware Mobile Service Discovery and Selection Mechanism using Artificial Neural Networks," ICEC, pp. 594-598, 2006.

[20] X. Dong, A. Halevy, J. Madhavan, E. Nemes, and J. Zhang, "Similarity search for web services". In: The International Journal on Very Large Databases, 2004.

[21] F. Emekci, O. Sahin, D. Agrawal, and A. Abbadi, "A peer-to-peer framework for web service discovery with ranking," ICWS, pp.192-199, 2004.

[22] E. Maximilien and M. Singh, "Conceptual model of Web service reputation," ACM SIGMOD Record, 31(4), 2002.

[23] M. Quzzani, "Efficient delivery of web services," PhD Thesis, Virginia Polytechnic, 2004.

[24] UDDI Version 3.0.2 Specifications, October 2004, http://uddi.org/pubs/uddi_v3.htm, Accessed February 2008.

[25] F. Hartman and H. Reynolds, "Was the universal service registry a dream?", Web Services Journal, December 2004.

[26] Google SOAP Search APIs, code.google.com/apis/soapsearch, Accessed February 2008.

[27] E. Al-Masri, and Q. H. Mahmoud, "Discovering the best web service," 16th International World Wide Web Conference (WWW), pp. 1257-1258, 2007.

[28] Y. Li, Y. Liu, L. Zhang, G. Li, B. Xie, and J. Sun, "An Exploratory Study of Web Services on the Internet," ICWS, pp. 380-387, 2007.

[29] Alexa Inc., Global Top 500, http://www.alexa.com/site/ds/top_sites?ts_mode=global, Accessed February 2008.

[30] J. Cho, H. Garcia-Molina, T. Haveliwalia, W. Lam, A. Paepcke, S. Raghavan, and G. Wesley, "Stanford WebBase Components and Applications," ACM Transactions on Internet Technology, Vol. 6, No. 2, pp. 153-186, 2006.

[31] C. Atkinson, P. Bostan, O. Hummel, D. Stoll, "A Practical Approach to Web Service Discovery and Retrieval," ICWS, pp. 241-248. 2007.