Crawling Multiple UDDI Business Registries

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

ABSTRACT

As Web services proliferate, size and magnitude of UDDI Business Registries (UBRs) are likely to increase. The ability to discover Web services of interest then across multiple UBRs becomes a major challenge specially when using primitive search methods provided by existing UDDI APIs. Clients do not have the time to endlessly search accessible UBRs for finding appropriate services particularly when operating via mobile devices. Finding services of interest should be time effective and highly productive. This paper addresses issues relating to the efficient access and discovery of Web services across multiple UBRs and introduces a novel exploration engine, the Web Service Crawler Engine (WSCE). WSCE is capable of crawling multiple UBRs, and enables for the establishment of a centralized Web services repository that can be used for discovering Web services much more efficiently. The paper presents experimental validation, results, and analysis of the proposed ideas.

Categories and Subject Descriptors

D.2.12 **[Software Engineering]**: Interoperability – *data mapping, distributed objects, interface definition languages;* H.3.5 **[Information Storage and Retrieval]**: Online Information Services – *data sharing, Web-based services*

General Terms: Design, Management, Measurement, Performance, Reliability, Verification

Keywords: UDDI, UDDI Business Registries, Crawler, Web Services, Discovery

1. INTRODUCTION

Web Services are Internet-based, modular applications that are becoming an emerging technology of choice for building understandable applications and are of an immense interest to governments, businesses, as well as individuals. As Web services proliferate, the same dilemma perceived in the discovery of Web pages will become tangible and the ability to search for a specific business or service will be time consuming particularly as the number of UDDI Business Registries (UBRs) begins to multiply.

In addition to that, having decentralized UBRs adds to the already existing complexity of how to effectively discover Web services. This is evident as new operating systems, applications, and APIs are equipped with built-in functionalities or tools for allowing businesses or organizations to create their own internal UBRs for intranet or extranet use such as Enterprise UDDI Services in Windows Server 2003, WebSphere Application Server, Systinet Business Service Registry, jUDDI, and among many others. Enabling organizations to self-operate and manage their own UBRs will maximize the likelihood of having a significant increase in the number of business registries and therefore, clients will soon face the challenge of finding relevant Web services across hundreds, if not thousands, of UBRs.

Although there have been numerous efforts that attempted to enhance the discovery of Web services [1,2], many of them failed to address the issue of handling discovery operations across multiple UBRs. To address the above issues, this work introduces a framework that extends our Web Service Repository Builder (WSRB) architecture [4] by enhancing the discovery of Web services without having any modifications to existing standards. In this paper, we propose the Web Service Crawler Engine (WSCE) which actively crawls accessible UBRs. Our solution has been tested and results show high performance rate when compared with other existing models.

WWW 2007, May 8–12, 2007, Banff, Alberta, Canada.
ACM 978-1-59593-654-7/07/0005.

2. MOTIVATIONS FOR WSCE

The crucial design of WSCE is motivated by several factors including: (1) the inability to periodically keep track of business and Web service life-cycle using existing UDDI design, which can provide extremely helpful information serving as the basis for documenting Web services across stages; (2) the inherent search criterion offered by UDDI inquiry API which would not be beneficial for finding services of interest; (3) the apparent disconnection between UBRs and the existing Web; and (4) performance issues with real-time search queries across multiple UBRs which will eventually become very time consuming as the number of UBRs increase while UDDI clients may not have the potential of searching every accessible UBR. Other factors of motivation will become apparent as we introduce WSCE.

3. WEB SERVICE CRAWLER ENGINE

WSCE is part of the Web Service Repository Builder (WSRB) in which it actively crawls accessible UBRs, and collects information in a centralized repository called the Web Service Storage (WSS). A Query Engine (QE) within WSRB provides clients with an interface to perform advanced search and discovery operations. The proposed discovery model that contains WSCE is shown on Figure 1.

Our approach in implementing the conceptual discovery model shown on Figure 1 is a process-per-service design in which WSRB 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. The WSCE Ws Seed List contains hundreds or thousands of business keys, service keys, and corresponding UBR inquiry locations.

WSCE begins with a collection of Web services and loops through taking a Web service from WsToCrawl queue. WSCE

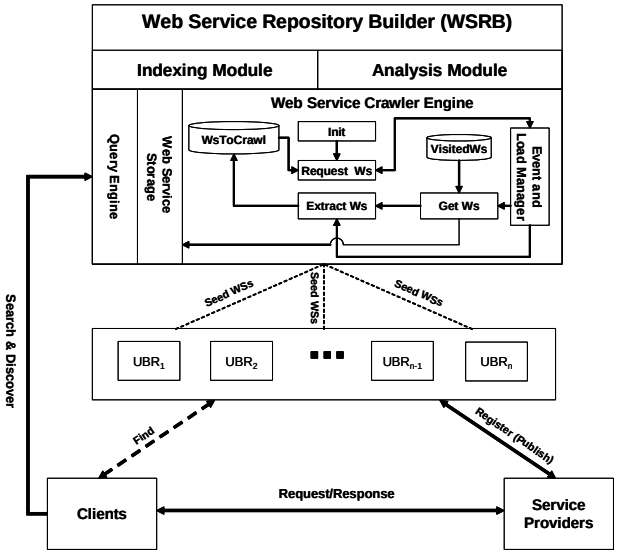

Figure 1. An Enhanced Discovery Model using WSCE.

then starts analyzing Web service information located within the registry, tModels, and any associated WSDL information through the Analysis Module. WSCE stores this information in the Web Service Storage (WSS) after processing it through the Indexing Module. After completion, WSCE adds an entry of the Web service (using serviceKey) into VisitedWs queue.

Conceptually, WSCE examines all Web services from accessible UBRs through businessKeys and serviceKeys and checks whether any new businessKeys or serviceKeys are extracted. If the businessKey or serviceKey has already been fetched, it is discarded; otherwise, it is added to the WsToCrawl queue. WSCE contains a queue of VisitedWS which includes a list of crawled Web services. In cases the crawler process fails or crashes, information is lost, and therefore, ELM handles such scenarios and updates the WsToCrawl through the Extract Ws component.

4. EXPERIMENTS AND RESULTS

Data used in this work are based on actual implementations of existing UBRs including: Microsoft, Microsoft Test, XMethods.net, and SAP. To compare performance of existing UBRs to WSRB, we measured the average time when performing search queries. The ratio has a direct effect on measurements since each UBR contains different number of Web services published. Therefore, the top 10% of the dataset matched is used.

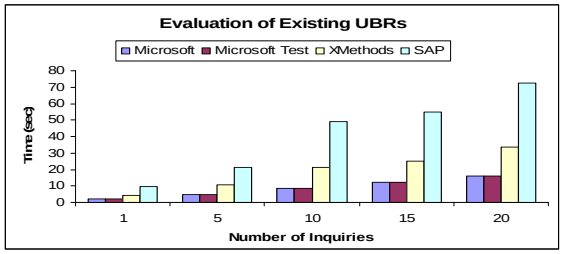

Figure 2. Evaluating Existing UBRs.

Figure 2 presents all search times for existing UBRs and demonstrates the fact that as the number of inquiries increases, the time increases significantly. For example, an inquiry to SAP UBR takes 9.7 seconds. Results presented on Figure 2 show the total times for an average query and exclude time taken for sending information over the network (network lag), time taken by operating system (system time), and time taken by program running the test (program time). Results from repeating the same test with WSRB are shown on Table 1.

Table 1. Results from running WSRB

# inquiries	1	5	10	15	20
WSRB Time (sec)	0.121	0.127	0.134	0.146	0.151

Table 1 results demonstrate significance and effectiveness of having WSCE via WSRB when compared to results shown on Figure 2. Based on these findings, querying multiple UBRs results in significant performance degrades while having a centralized framework such as WSCE via WSRB improves performance rates tremendously. In order to measure the efficiency of our approach, another test was conducted by performing a search query to all UBRs concurrently and measuring the total time it takes to obtain the top 10% of the matching dataset. Performance results from this test are compared with WSRB on Table 2.

Table 2. Comparison of performance of WSRB vs. all UBRs

# inquiries	1	5	10
All UBRs Time (sec)	16.920	64.140	97.100
WSRB Time (sec)	0.121	0.127	0.134
Inquiry Time Ratio	140	505	725

Table 2 demonstrates that conducting a single query to multiple UBRs, for example, takes approximately 16.92 seconds to receive a response which may not be practical particularly if clients are searching for Web services via mobile devices. In addition, the inquiry time ratio between WSRB and multiple UBRs increases significantly as the number of concurrent queries increases

5. CONCLUSION

A Web Service Crawler Engine (WSCE) has been presented in this paper for the purpose of effectively discovering Web services. The proposed solution provides an efficient Web service discovery model in which clients do not have to endlessly search existing UBRs for finding services of interest. As the number of Web services increase, the success of businesses will depend on service discovery and performance time when searching multiple UBRs. Our experiments demonstrate that building a crawler and a centralized repository for Web services is inevitable. For future work, we plan to extend our current framework to include a ranking mechanism that outputs desired services of interest within top results and therefore, rendering the discovery process to become more efficient.

6. REFERENCES

[1] E. Maximilien and M. Singh. Conceptual Model of Web Service Reputation. ACM SIGMOD Record, 31(4), 2002.

[2] K. Sivashanmugam, K. Verma, and A. Sheth, Discovery of Web Services in a Federated Registry Environment, Proceedings of IEEE ICWS, pp. 270-278, 2004.

[3] 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), 2007.