GHSOM-Based Web Service Discovery

FLORIE ISMAILI¹, BOGDAN SHISHEDJIEV², XHEMAL ZENUNI³, BUJAR RAUFİ⁴
Faculty of Contemporary Sciences and Technologies¹,³,⁴, Faculty of Electronic Engineering and Technology²
South East European University¹,³,⁴, Technical University of Sofia²
Campus Bldg.305.20, Ilindenska nn,1200¹,³,⁴, Tetovo Kliment Ohridski St.Sofia-1000²
FYROM¹,³,⁴, BULGARIA²

f.ismaili@seeu.edu.mk¹, xh.zenuni@seeu.edu.mk³, b.raufi@seeu.edu.mk⁴, Bogi@tu-sofia.bg²

Abstract: - The increasing number of available web services necessitates new ways for efficient Web service discovery. In this paper we follow up our previous work of proposing new framework for efficient web service discovery by combining the growing hierarchical self-organizing map (GHSOM), a powerful data mining technique for the clustering and visualization of large and complex data sets with Latent semantic Indexing. This framework supports web service discovery and service ranking by giving an ordered list of relevant services which helps users to make better decisions in selecting the right Web Service for reuse.

A prototype system implementation is provided to show the validity of this framework. In order to achieve this, the system uses and extends existing web service tools.

Key-Words: - Web services, Web service discovery, growing hierarchical self-organizing (GHSOM), data mining, service ranking, Latent Semantic Indexing.

1 Introduction

Web Services are changing the Web by being adopted in different applications starting from Business-to-Business to Business-to-Consumer applications. Web services can be interoperable due to its supporting technology XML, SOAP and WSDL [5]. Without publishing Web services through registries, service requestors will not be able to locate them, and service providers will be forced to find other channels for publishing their services.

Web Services supports UDDI for service discovery. Web service discovery mechanism is finding cost effective and efficient list of services that meet user-specified functional requirements. Even UDDI has many features for Web service discovery, it still lacks of limitations in its search mechanism because of yielding in coarse results [6].

Search engines such as Google, AlltheWeb, Yahoo and Baidu have become a new source for finding Web services. However, search engines generally crawl Web pages from accessible Web sites and fetch WSDL documents residing on Web servers. But still this is publishing service information without analyzing the basic service properties like binding information, operations, ports etc.

In recent years, other trends for finding services have also emerged, Web service portals as WebServiceList, RemoteMethods, WSIndex, and XMethods.net, but they and can easily become disconnected from the Web service environments.

Based on the above, in this paper we will discuss in detail the support for Web service discovery with SSD[2] and its implementation details. The decision to design and implement a new tool for Web service discovery was based on the detailed analysis of Web service research challenges and open issues [1, 2].

Based on our research for Web Service Discovery we identified the following requirements our software system should meet in order to fulfill our purposes:

- Be able to pre-process large collections of data with a flexible set of available pre-processing techniques.
- Easy composition of various text pre-processing and mining methods.
- Efficient ranking

The rest of the paper is organized as follows: we first discuss the related work done in the area of Web service discovery, in section 3 we briefly introduce our Service Discovery Architecture called SSD Layer followed by description of detailed implementation of the system and presentation of the results in section 4. Future work is given in section 5 and finally we conclude.
2 Related Work

Discovery of Web Services is a fundamental research challenge in Service Oriented Architecture. It involves the process of discovering, selecting and executing WS to achieve the objective of a user request. This section reviews features of other efforts for web service discovery.

Web-based directories such as XMethod, RemoteMethods or WebServiceList are simple service portals which gave the ability for searching Web services based on a keyword search paradigm of Web service descriptions. But, here the metadata is not well organized, as a result Web service data will not be easily discovered and therefore, results will not yield to effectively selected services.

Meteor-S [7] presents a framework for enhancing UDDI with semantics and having multiple public/private registries grouped into registry federations. However, in my opinion it still does not provide any possibility for advanced search techniques for finding specific business applications.

Some approaches focused on the peer-to-peer framework architecture for service discovery and ranking [8]. They provide keyword-based search engine for querying Web Services which actually are very limited set of search capabilities that makes hard for the clients to formulate their service queries tailored to their needs.

Other approaches are proposing Service Discovery and Reconfiguration Framework at runtime that suggest to deploy and reconfigure the software which makes it possible to add, remove, and replace services during running. However to do that, the user should have a prior knowledge of exact input and output parameters [9].

Differently of these service discovery approaches, this paper presents a new approach for Web service discovery which we believe that is the first attempt that combines GHSOM clustering technique with Latent Semantic Indexing, where the proposed method improves the quality of service discovery.

3 Overview of Service Discovery - SSD Architecture

In our previous works [1, 2] we have proposed a new framework for Web services discovery and we have outlined the architecture of our service discovery approach.

Web service discovery process begins with taking as input a user query, which is used as search criterion and returns a set of web services matching this query.

The process of discovery consists of the following phases:

- A register server will be crated, so that all the web services which will be collected from UDDI, Service Portals, and Search Engines will be stored at this register.
- The searching process will be initialized with a query typed by the user which will be used as search criteria.
- Applying GHSOM to the service set, in order to construct a self organizing semantic map for service retrieval.
- Utilizing Latent Semantic Indexing for scoring and ranking the documents by their relevance.
- Returning the achieved result in the user interface.
4 Prototype System Implementation Details
The system is based on GHSOM algorithm, designed and implemented using a pair of GHSOM neural networks to create hierarchical maps of the WSDL files. The HTML based visualization is provided that allows users to see their Web Services of interest. A user interface is created which ranks WSDL files based on user requirements.

![Data Flow Diagram]

4.1 Data Set
The Local Registry of Web Service Description file is created by collecting WSDL files from different sources, starting from UDDI, several Web Service portals as WebServiceX.net, WebServicesList, RemoteMethods, WSIndex, XMethods.net as well as from search engines such as Google an Yahoo.

4.2 Data Preprocessing
The A “Transformer” program is developed. The Transformer reads all WSDL files from Local Registry, extracts important information like “type name”, “message name”, “service name” and “documentations” using a XSLT script. All this information is converted to corresponding text files. The transformer than will continue with preprocessing, including eliminating irrelevant information like reading two <documentation> from single WSDL file, stop words elimination, stemming and indexing keywords in order to obtain high-quality features for describing WSDL files. Here is important to note that some features of the SOMLib Java package are used to create the \( \text{tf x idf} \) input vectors.

The \( \text{tf x idf} \) weight vector (1) is calculated by multiplying the term frequency by the inverse document frequency (Salton, 1989) and the word length of term for each document [3].

\[
w_i(d) = tf_i \times \log \left( \frac{N}{df_i} \right)
\]  

where \( w_i(d) \) represents the weight of term (i) in document (d), \( tf_i \) represents the number of times term (i) appears in document (d), \( N \) represents the total number of documents, \( df_i \) represents how many documents contain term (i).

4.3 Generating Maps
The most time consuming was finding a way to represent the data as weighted input vector required by GHSOM. For GHSOM training is used the C++ implementation of GHSOM algorithm developed by Ruber, Merkl and Dittenbanch with very little modifications.

![GHSOM Architecture Diagram]
The GHSOM algorithm takes its input from a flat text files using DataTraining cmdlet, and it outputs to XHTML maps, which then are serialized in XML format used by the system for further development.

In the resulting map architecture the WSDL files are grouped into various topical branches, with topics in each branch being arranged on two-dimensional maps and keywords serving as labels for the respective topics. Each hierarchy presents a group of the services related according to their semantic similarity.

![Layer 2 Map](image)

Fig.4, Results of the Second Layer Map

3.4 Service Ranking
Finally the WS-Ranking cmdlet is developed, which loads the maps generated by GHSOM, calculates the user requirements and produces an output to the User Interface, which is a list of ranked relevant services.

Also a user friendly WS-Testing interface is provided, accessed by a link near each of returned WSDL file. This is an extended version of WsdlReader. They can test the Web Service, by giving input parameter values through interface, and see the result on the same interface, but it supports only simple type of input and output parameters. The information about the Web Service is taken from the WSDL file (Web service endpoint, methods, input and output parameters) by producing a WebRequest and sends it to the WSDL.

5 Future Work

- Enhancing Semantic Web Service based discovery- performing searching and ranking using detailed semantic information including matching degrees between concepts in ontology.
- Extending of text-mining services with semantic description and making them available as a set of semantic Web services which can be used by other distributed applications that can express their needs in a form of appropriate ontology.
- System Evaluation

6 Conclusion
In this paper we presented the implementation of our Semantically Service Discovery- SSD Architecture and identified future research directions. We demonstrated the potential of the GHSOM for organization and visualization of WSDL file repository. Furthermore, we proved that the proposed solution provides an efficient Web Service discovery model and sketched how the system can be extended as well as what advantages that such kind of implementation could bring.

References:


