Chapter 4

An Efficient Approach for Dynamic Web Service Selection in Service-Oriented Architecture

4.1 Introduction

Service-Oriented Computing (SOC) promises a world of cooperating services loosely connected, creating dynamic business processes and agile applications that span organizations and platforms. The Service-Oriented Computing paradigm uses service to support the development of rapid, low-cost, interoperable, evolvable and massively distributed applications. Services are autonomous, platform-independent entities that can be described, published, discovered and loosely coupled. Realizing the SOC promise requires the design of Service-Oriented Architectures that enable the development of simpler and cheaper distributed applications. Now-a-days Service Oriented Architecture (SOA) [1-2] is very relevant prototype in enterprise architectures and information technology systems. Service Oriented Architecture provides a flexible, robust and dynamic pattern for integrating distributed services into business processes.

Web Service is modular, self-describing and loosely coupled software application as described in [3]. Web service is comprised of software modules that describe compilation of operations and network-accessible through standardized eXtensible Markup Language (XML) messaging. It provides mean for realizing reliable, secure, scalable, interoperable distributed and service-oriented software system using the web on the basis of well defined and establishment standards. For this reason, they have been broadly used in information technology industry in recent years. Web services are useful computational abstraction because the interfaces to the services are well-specified by standard languages, web services and their users can be written in different languages. Web service selection is the major issue in modern SOA system.
The main objective of web service selection is to select an optimal web service for the current task as per demand of the user. The following number of key questionnaires is practical to consider for describing the term optimal:

i) What kind of information is needed in the selection process?
ii) At what position should a selection mechanism be used?
iii) How can this information be acquired?
iv) How will this information be used in the selection process?

These above questionnaires provide greater freedom for both the service owner and software developer. Using a set of standard technologies such as Simple Object Access Protocol (SOAP), Web Service Description Language [4] and UDDI, a lot of web services can be accessed and invoked across the web. Researchers have approached the first question from a number of different angles. As an example, Aldo de Moor et al. looked at the role of web service selection during the software development cycle [5]. In virtual communities, they argued, web service selection should not be left to the software developer alone, in lieu of that it is important to involve members of the community to understand their requirements and involve them in the process of selection.

In general, web services interact in three primary modes: service publishing, finding and binding. Interactions depend on the web service artifacts, which include the service implementation and the service description. Selection of particular web service among the web services is not an easy task as per the goal of user. To discover the exact web service, semantic are very much essential. Moreover, Ontologies have been used to provide meaning to every term of web service description model. They serve twofold purpose: i) defining the formal semantic of the information ii) linking machine and human terminologies. Semantic plays an important role to discover the web service automatically.

Automatic discovery of web services cannot be achieved by a conceptual model alone. Web Service Modeling Ontology (WSMO) [6] has been applied to conceptualize the
service objectives. Web Service Modeling Language (WSML) has been used to formalize WSMO. Additionally, jUDDI (open source software based on java technology) has been utilized as repository to store web services.

Generally, Web services have two main parts: i) capability means (Functionalities offered by a web service) ii) interface (how these functionalities can be achieved). Moreover, Goals are used to express consumer’s desire in WSMO.

4.2 Different Existing Approaches for Web Service Selection

There are various types of web service selection method exist in the service oriented computing world. The major purpose of web service selection is to select the exact web service in optimal way for the current task. At present, the modern web service architecture and semantic web efforts address the problem of web service discovery rather web service selection mechanism. Service discovery method provides set of e-services those correspond to a predetermined user request while selection deals with choosing a service between those are discovered against the user request. The purpose of service discovery is matching a user request with web service functionalities. Moreover, selection seems is the most important problem what we are dealing with. In fact, if the discovery process is exhaustive, a very large number of services may be found. Selection process is very much harder due to the huge number of e-services and candidate services.

Dumitru Roman et al [7] illustrated service oriented architecture in order to express web services. Even though more functionality is incorporated into service descriptions, it still remains tricky for selection to discover the subset of services that will be part of the composition. Though functional attributes have been integrated by web service architecture, selection should consider more than functional criteria to make a distinction between discovered services. They use the Quality of Service (QoS) model, which composes of time, cost, availability and reputation as non-functional criteria. Since non-functional criteria have been included by each service, selection can use these QoS variables in order to select the optimal subset from the entire discovered services.
A lot of literatures have been demonstrated and different methodologies are depicted for the solution of the dynamic web service selection in the service oriented framework. Some studies on web service selection have been described here. Liu et al [8] used the concept of adjacency matrix in the web service selection process, where output of one web service is matched with input of other web services. On the other hand, Aydogan et al [9] showed the dependency graph structures to resolve web service composition problem. Xiaoping Ma et al [10] discussed about the AND/OR tree search method based on advance effort of artificial intelligence, but it suffered from huge amount of calculation.

Y. Charif et al [11] proposed a dynamic mechanism for performing web service selection through unplanned agent based dialogue. E. Maxilien et al [12] offered a multi agent based architecture. They applied the semantic web technology in order to select the best service according to requestor’s needs. Additionally, they exemplified an architecture that consists of interacting agents. The authors also illustrate that the every web service had an autonomous agent connect to it, which offered the same interface as the service. This interface permitted the service requestor and agent to communicate, interact and share information through agencies.

Liu et al [13] put forward an procedure for major selection criteria which were depended on the QoS based service selection. Moreover, the authors had measured three quality criteria namely execution time, execution duration and reputation for the dynamic selection of web services. They suggested an open, reasonable and dynamic framework that evaluated the QoS of the available web services by using consumers’ response.

Demian Antony et al [14] introduced the QoS broker oriented web service architecture to select the best web service for the service requester. They also explored different types of requester’s QoS constraints and suggested a web service selection method which was defined on the tree model of QoS constraint.
M. Adel Serhani et al [15] offered a QoS broker based architecture for web service selection. Goal of the broker was also discussed in their research works. The goal supported web services QoS verification, certification, confirmation, selection and monitoring. They elaborately described the key features of the broker that was not supported by existing approaches. The main contribution of their work was the design of the broker that could be invoked by interested requesters when developed and published as a web service. The authors emphasized more on the verification and certification process and employed a methodological mechanism to measure the QoS attributes. Furthermore, the authors illustrated the applicability of the architecture’s roles with prototype implementation.

Day et al. specified a dynamic method of web service selection that was based on QoS information [16]. This approach employed a centralized forum system to maintain track of the entire data and marked up interactions between service requestor and web service, thereby allowing service requestor to reason over and select from a number of prospective services. The authors in [17] formulated the dynamic web service selection problem in a dynamic and failure-prone environment. They exploited an FSM to model the invocation order of operations in each web service and constructed a web service composition that enumerates all possible delegations. In addition to that, authors showed that the aggregated reliabilities could be seen as the eigenvector of a square matrix with Eigen value being 1. They also proposed an iterative power method to efficiently compute them.

Balke et al [18] described a system whereby a user’s goal influences the selection of the web service. A SQL-like query language has been used to narrow down the potential service. Any services that did not permit querying with the user’s terms were discarded. The authors described this one as “hard constraints.” If the user had any preferences, these were extracted from user profile, and used to even further narrow down the numbers of services these were define as “implicit soft constraints”.
Selection of an appropriate web service for a particular user goal has become a difficult challenge due to the increasing and huge number of web services offering similar functionalities. When dynamic discovery mechanism is used to select a specific service in optimal way, it is common that the result of the discovery contains more and supplementary than one provider. From this bulkiness and massiveness, it is tough enough even for a sophisticated user to find out the matter of interest due to syntactical match based searching. We have tried to impart semantic to the offered web services so that the response to the consumer might be more specific to select the exact service.

In this chapter, we have elaborately discussed a novel concept of matching of semantically enriched web services with the user’s goal. Instead of invoking different sets of web services, the system will generate only one of the best choices available in the service repository. The dynamic web service selection through the graph theory approach in service-oriented environment is the main concern of this proposed work. The proposed algorithm has been evaluated in e-governance domain. In addition to that we have compared this algorithm with the various existing approaches.

4.3 Problem Definition and Propose Dynamic Web Service Selection in Service Oriented Architecture

Here, a framework for dynamic web service selection has been proposed. This framework has been broadly decomposed into the following six major components

i) Graphical User Interface

ii) Sub-Goal Generation Module

iii) Syntactical Selection of Web Service

iv) Semantically Selection of Web Service

v) Matrix Generation

vi) Selected Set of Web Services.

The detailed workflow of the general architecture of the proposed model has been shown in following Figure 4.1. The sequence of interactions among the components has been
depicted here for the clarification purpose. Every module has its unique activities in this architecture to provide the exact web service as per user need in optimal manner.

The following activities take place one after another while a user submits composite goal to obtain the desired services in service-oriented architecture:

1. At the very outset, service requestor enters the composite queries/goals in the entry form of portal through Graphical User Interface (GUI) for receiving exact web service as per their requirement.

2. Subsequently, Sub-Goal Generation Module formalizes and decomposes requestor’s composite goal into one or several independent sub goals to reduce the complexity.

3. Afterward, syntactic matching of web service with the independent sub-goal has been performed by Syntactical Matching module.

Figure 4.1 Architecture for Dynamic Web Service Selection with Sequence of Interactions
Consequently, sub-goals are sent for semantic matching.

Syntactically matched web services are selected for semantically matching.

Semantically matched web services are sent to selection matrix generator.

Select and compose of web services.

Finally, exact web service is sent to the user.

The above mentioned flows of interactions are very much constructive for the inexperienced user to acquire the appropriate service as per their need. It automatically generates the desired result for the user using the semantic. Each module has their unique features and activities for providing the most excellent result.

4.4 Proposed Algorithms for Dynamic Web Service Selection

The major reason of dynamic web service selection is to choose exact web service in optimal way. When self-motivated discovery is used in web service, it is widespread that the result of the discovery contains more than one provider. Even for a merged web service consists of many atomic web services and the selection issue still needs to be addressed when there are multiple providers available for an atomic service. In order to formulate a distinction between the services which provide the same functionality, selection criteria should be used. They facilitate to evaluate the web services within a group and choose the component that matches the need and the preference of the requestors while taking into account the abilities of the providers.

4.4.1 Overview of the Proposed Web Service Selection Mechanism

Here, graph theory has been used in the proposed web service selection method. A graph is a representation of a set of objects where some pairs of objects are connected by links. The interconnected objects are represented by mathematical abstractions called vertices, and the links that connect some pairs of vertices are called edges. Vertices are also called
nodes or points, and edges are also called arcs or lines. A graph can be represented by matrix. In this chapter, we consider user’s goal as vertex and web service in the position of edges for the service oriented architecture.

We consider a composite goal $G_n$ which divides into $n$ sub goals for matching with the web service to get exact service. Let there are $m$ numbers of web services stored in the jUDDI repository for future use.

We define an $n \times m$ matrix $A = [a_{ij}]$ whose $n$ rows correspond to $n$ goals and $m$ columns correspond to $m$ web services:

The matrix elements $a_{ij} = 1$, if $i^{th}$ goal is satisfied by $j^{th}$ web service $= 0$, otherwise.

What we want to say that if a goal is satisfied or matched by a web service, then we assign a value 1 at the intersection of corresponding row and column. If there is no match found then it assign to 0. This is a simple binary matrix of goals and web services. If there is 1 under a web service for a goal, we articulate that the web service has been discovered against that particular goal. More than one web services may be discovered for a single goal.

The following matrix has been derived from the above mentioned pre-defined logic:

<table>
<thead>
<tr>
<th></th>
<th>$W_1$</th>
<th>$W_2$</th>
<th>$W_3$</th>
<th>..........</th>
<th>$W_{m-1}$</th>
<th>$W_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G_1$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>..........</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$G_2$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>..........</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$G_3$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>..........</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>......</td>
<td>......</td>
<td>......</td>
<td>......</td>
<td>..........</td>
<td>..........</td>
<td>......</td>
</tr>
<tr>
<td>$G_n$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>..........</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.1 Matrix of Web Services and Goals
In the aforementioned Table 4.1, the intersection of \(G_1\) (Sub Goal) and \(W_2\) (Web Service) is 1. This implies that the user’s goal \(G_1\) is satisfied by \(W_2\). In case there is no 1 in a particular row, it is concluded that the particular goal is not satisfied with any other web service i.e. either the goal is not properly defined or web service registry is not well equipped as per that goal. Therefore it is implied that 1 indicates the corresponding goal has been achieved.

We count number of 1’s of each column and store it in a counter. The web service with maximum number of 1 is selected first. The row index of 0’s in that column has been identified. The corresponding goals are still unachieved. Now, the particular web service is detected to cover most of these unsatisfied goals. Then this web service is selected. If there is still unsatisfied goal (i.e. there is still 0) in the column of first selected web service, we will search for the web service which will satisfy most of these 0’s and selected next. Thus when all the 0’s of the column of first selected web service is satisfied by other web services, we stop the searching. The set of selected web services is composed and sent to the user.

The following matrix has been considered for illustration the scenario:

<table>
<thead>
<tr>
<th></th>
<th>(W_1)</th>
<th>(W_2)</th>
<th>(W_3)</th>
<th>(W_4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(G_1)</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(G_2)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(G_3)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(G_4)</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

In this procedure, maximum number of 1 is two, which corresponds to the column of \(W_1\). So \(W_1\) is selected undoubtedly. But two goals \(G_2\) and \(G_3\) are not satisfied by \(W_1\). None of the web services can satisfy these two goals simultaneously. \(G_2\) is achieved through \(W_2\). Hence we select \(W_2\). Now we found match for \(G_3\). From the table, we found that \(G_3\) is achieved through \(W_4\). So \(W_4\) is selected. These three web services together (\(W_1\), \(W_2\) and \(W_4\)) form a set. This set is sent to the user.
4.4.2 Algorithm for Web Service Selection

The following algorithm shows the necessary steps to select exact web service and acquires the maximum quality results. For finding a service for a specified task, performs a search on web service descriptions as well as goal ontologies.

**Input:** Goal Ontologies, Web Service Description (Ontology).

**Output:** Single set of selected web service.

**Prerequisites:**

- \( G_n \): Represents sub goals of a composite goal where \( n \geq 1 \).
- \( W_m \): Web service(s) required to fulfill the goal where \( m \geq 1 \).
- \( S_{nxm} \): The selection matrix of dimension nxm where columns are represented by \( W_j \) (\( j=1 \) to \( m \)) and rows are represented by \( G_i \) (\( i=1 \) to \( n \)).

**Step 1:** for \( i=1 \) to \( n \) repeat steps 2 to 3.

**Step 2:** for \( j=1 \) to \( m \) repeat step 3.

**Step 3:**

- if precondition \( G_i \) is_subset_of precondition \( W_j \)
- and post condition \( G_i \) is_subset_of \( W_j \) then \( S_{ij}=1 \).
- Else
  - \( S_{ij}=0 \).

**Step 4:** Select that \( W_i \) in which maximum no of 1 is present.

**Step 5:** Identify the rows \( S_{ki} \) where \( S_{ki}=0 \) and \( 1 \leq k \leq n \). and \( i \) represents column number of the already selected web service.

**Step 6:** Select \( W_j \) (\( j \neq i \)) which has maximum number of 1 in the matrix against the rows \( S_{ki} \) containing 0.
**Step 7:** Add $W_j$ to the set called ‘Selected Web Service set’.

**Step 8:** Continue step 6 to 7 until all the unsatisfied goals ($S_{ij}=0$) are achieved.

**Step 9:** Execute the web services in ‘Selected Web Service’ to achieve the composite goal.

The matrix has been formed here to find the relations between service requestor goal and available web services in the repository.

Let there are $m$ web services stored in jUDDI repository to meet up user goal. Suppose that the probability of matching a goal with a web service is $p$. Then the number of edge formed will be $p \, C_n \times C_m$. Thus the time complexity of the aforesaid algorithm is $O (n^2)$ in the worst case. Most of the cases, the system generates much better result in less time.

### 4.5 Cycle Sequence Diagram of the Proposed Model

A naive user gives the necessary information in Graphical User Interface (GUI) based on HTML. If the goal of the user is composite then it divides into straightforward sub goals by the system. These sub goals are matched syntactically with available web services in the jUDDI repository. The selected web services are again passed through a phase of matchmaking with the simple sub goals, but this time semantically. The web services form a simple binary $(0, 1)$ matrix with the corresponding goals. The most excellent set of web services replies the queries of the client.

Sequence diagram is the most common kind of interaction diagram, which focuses on the message interchange between a numbers of lifelines. Sequence diagram describes an interaction by focusing on the sequence of messages that are exchanged, along with their corresponding occurrence specifications on the lifelines. It also describes an interaction by focusing on the sequence of messages that are exchanged, along with their corresponding occurrence specifications on the lifelines. A sequence diagram shows, as parallel vertical lines, different processes or objects that live simultaneously and as
horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

The cycle sequence diagram of the proposed model has been illustrated in the following Figure 4.2

![Diagram](image-url)

**Figure 4.2 Enrichment Cycle Sequence Diagram**

At the outset, service requestor submits the composite goal to the framework as per their requirement through Graphical User Interface (GUI).

1. The composite goal is alienated into sub-goals with the help of sub goal generation module.
2. Then web services are matched with goals syntactically.
3. Goals are sent to selection matrix generator.
4. Web services are sent for selection matrix generator to match with goals with the help of semantic.

5. Best possible set of web services are sent to user.

4.6 Usage Scenario and Discussions

We have selected real world services for the censorious evaluation of the proposed framework. The aforementioned model has been presented in Web Service Modeling Language for the sake of experiment. WSML-Flight language has been selected for the WSML implementation to provide more expressiveness. The Health System Ontology (HSO) using WSML has been developed.

In order to meet the censorious evaluation of the web service selection algorithm, we have been considered four web services and four goals for the sake of simplicity and straightforwardness. The web services are WSA, WSbloodsugar, WS4enrolment, and WS4doctor. All the four web services developed using WSDL format and published in jUDDI repository for discovery and selection purpose. The various ontologies for Health Service are also stored in the said registry.

Let the predefined goals are as follows: Goal1, Goal2, Goal2enrol, Goal2getdoctor. When goal is stored in the framework, WSbloodsugar is achieved after syntactical and semantically checking. Similarly, WSB has been obtained for Goal2, WS4enrolment is selected for Goal2enrol and WS4doctor has been achieved for Goal2getdoctor. In the corresponding cross-sections of Goals and web services, which are selected, denoted as 1 and other places are left as zero. After applying the proposed algorithm, we have achieved the best set of services in less time with better QoS. It provides a better and transparent service selection in terms of service requestors’ point of view. In the proposed work, the end user is free to express their requirements using Graphical User Interface (GUI) without any constraint to a general goal and predefined list.
4.7 Comparative Study between the Existing Method and the Proposed Method for Dynamic Web Service Selection

In this sub section, we have considered different popular mechanisms in this domain to build a comparative study with the proposed algorithm. The comparison has been deducted on some parameters like dynamic selection, transparency, reasoning mechanism and runtime selection.

The comparison is given below in Table 4.3

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic Web Service Selection</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Web Service Selection Transparency</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Consumer Involved in QoS Computation</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Clients involved in web services selection</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Goal based Selection</td>
</tr>
<tr>
<td>Clients give feedback</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Selection</td>
<td>Run-Time</td>
<td>Run-Time</td>
<td>Run-Time</td>
<td>Run-Time</td>
<td>Run-Time</td>
</tr>
</tbody>
</table>
4.8 Conclusion and Future Works

In practice, we have been able to solve some of the web service selection problems that were earlier considered as challenging for the existing techniques. However, there are still remaining some unsolved issues in this area. This includes transparent service selection, level of security, improving fault handling, addition of weights or fuzzy elements in the decision making system and extension of dynamic web service selection.

In this chapter, we propose a novel concept of web service selection where a service requestor can dynamically choose the web service from the service-oriented architecture in optimal manner. The major reason of dynamic web service selection is to choose exact web service in optimal way. A framework comprised of seven modules has been developed to select the exact web service. Besides, some new algorithms have been included. On receiving the composite goal from user, this framework splits the composite goal into sub goals. Graph theory has been used in this proposed mechanism. We consider user’s goal as vertex and web service in the position of edge for the service oriented architecture. Afterward a matrix has been formed to find the relations between service requestor’s goal and available web services in the repository. An algorithm has been applied on goal ontologies and web services to provide the exact service in optimal mode. Therefore, this framework assures the level of security, since the users do not have the direct access to the web services. It also provides transparent service selection from the user’s point of view. Four web services and goals have been developed in the domain of health system for evaluation of the framework and its web service selection algorithm. In near future, we have a plan to include fault handling mechanism in the proposed work.
References


[10] Xiaoping Ma, Baotian Dong and Ming He “And/OR Tree Search Algorithm in Web Service Composition” IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, 2008


