

# DYNAMIC SELECTION OF WEB SERVICES

Dr. Ilavarasan Egambaram

Department of Computer Science & Engineering, Pondicherry Engineering College, Pondicherry, India  
eilavarasan@pec.edu

G. Vadivelou

Department of Computer Science, Kanchi Mamunivar Centre for PG Studies, Puducherry, India  
vadi\_ganesh@hotmail.com

S. Prasath Sivasubramanian

Department of Computer Science, Avvaiyar Govt. College for Women, Karaikal, Puducherry, India  
prasathsiva@hotmail.com

## ABSTRACT

Web Service is a software component invoked over the Web via an XML message that follows the Simple object Access Protocol (SOAP), which is a simple XML based protocol to let applications exchange information over HTTP and to transport the messages using open protocols standard. Web Services are based on distributed technology and provide standard means of interoperating between different software applications across and within organizational boundaries with the use of XML. Web Services technologies allow interaction between applications. Sometimes a single service given alone does not meet user's needs. In this case, it is necessary to compose several services in order to achieve the user's goal. The composition of web services could be static or dynamic. The differentiation between static and dynamic composition deals with the point of time at which a concrete Web Service is integrated into the specification of a composition. With static composition the concrete services are determined and integrated into the specification at design-time. With dynamic composition on the other hand, at design-time there is only a specification of the type of service given. The concrete service is then integrated at run-time. Thereby it is possible that the concrete service has to be discovered first or that it is already known at run-time. The purpose of web service selection is to select optimal web service for a particular task. When dynamic discovery is used in Web Services, it is common that the result of the discovery contains more than one provider. The purpose of this work is to develop an architecture for secured, dynamic and transparent service selection and to evaluate the proposed architecture in terms of what selection techniques should be applied.

**Keywords:** QoS, Security, SOAP, Web Services.

## 1. INTRODUCTION

The Service Oriented Architecture(SOA) allows the development of modular components that can be dynamically discovered and incorporated into platform and language independent applications, using just-in-time integration [1] [2] [3]. The Web Service is the most popular technology which evolves from the SOA paradigm. The basic Web Service platform is combination of HTTP and XML[4]. The HTTP protocol is the commonly used internet protocol. XML provides a language which can be used between different platforms and

programming languages. Web services use XML to code and to decode data, and SOAP[5] for establishing sessions to exchange information. Web Services are web applications whose interfaces are exposed over protocols like HTML and XML[6]. Web Services are described by Web Service Definition Language (WSDL) in XML format. WSDL [7] is a major language that provides a model and an XML format to describe the syntax about Web services. It acts as a vocabulary, associated with UDDI[8]. The Web Service description hides the implementation details, but at

the same time, gives enough information that is necessary for the service interaction.

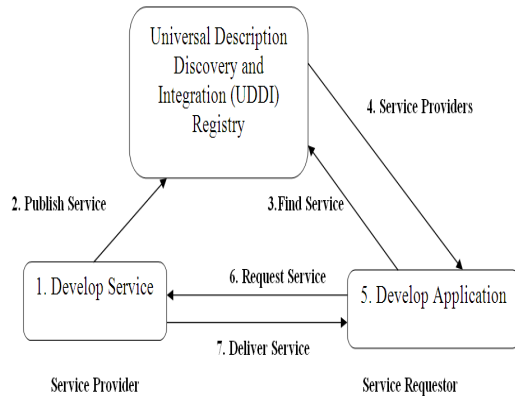


Figure 1.2: Web Service Design Paradigm

The Web Service Design paradigm consists of three phases which is shown in the Figure 1.2. The model begins with the publish phase, when an organization decides to offer a Web Service. The Web Service can be an existing application with a new Web Service front end, or it can be a totally new application. Once an enterprise has developed the application and made it available as a Web Service, the enterprise describes the interface to the application so that potential users interested in subscribing to it can understand how to access it. This description can be oral, in some human language such as English, or it can be in a form, such as WSDL, that can be understood by Web Services development tools. To facilitate automated lookups, the service provider advertises the existence of the service by publishing it in a registry.

The next step of the model is the find phase. Once the service is advertised in a UDDI registry, potential subscribers can search for possible providers and implement applications that utilize the service. Potential subscribers use the entries in the registry to learn about the company offering the service, the service being offered, and the interface to the service. The final phase of the model is the bind phase. When a subscriber decides to use a published service, it must implement the service interface, also called binding to the service, and negotiate with the service provider for the use of the service. When the application has been implemented and the business relationships resolved, the Web Service is utilized operationally. The only participants at this point are the service subscriber, who requests the service, and the service provider, who delivers the service. WSDL and UDDI registries are generally only used during

the initial discovery of the service and the design of the application.

Web Services can encapsulate a specific task or can be designed as a composition of other services, representing a complex aggregation. Service providers describe their Web Services and advertise them in a universal registry called UDDI. This enables service requestors to search the registry and find services. UDDI allows for the creation of registries that are accessible over the web. Once the list of services, offering the same functionality, is available and the criteria are specified, a set of steps (instructions, rules) for Web Service selection should be followed during the decision making process, in order to determine the component to handle the client's request. Various sets of instructions represent different selection techniques.

The rest of this paper is structured as follows: In Section 2, we discuss about the motivation for this work. In Section 3, the problem of dynamic WS selection is defined and then we describe our approach to dynamic WS selection.. Finally, the conclusion and future enhancements are given in Section 4.

## 2. MOTIVATION

Quality of Service is an aggregated metric for describing characteristics of systems in areas, such as networks and distributed systems [9]. Ran [9] discusses that although international quality standards are available for some domains (such as Software Engineering), for others, including Web Services, there is no consensus about the QoS measures that should be taken into account. Researchers propose various ways for grouping the QoS metrics into different categories. Liu, Ngu, and Zeng [5] present the QoS measures as generic or domain specific. The first group includes features, common for every service, such as price, execution time, dependability, and failure rate, whereas the second group refers to criteria, specific to a particular domain. In addition, the paper provides another grouping of the QoS metrics - deterministic and non-deterministic. The deterministic group includes features that are known at the time when the service is invoked (such as price), whereas the non-deterministic group consists of criteria that are uncertain when the Web Service is called (such as response time). Maximilien and Singh [10] divide the QoS attributes into objective and subjective. The former include QoS features such as availability, reliability, and response time, and the latter refers to the clients' experience. Ran [9] proposes another classification of the QoS

QoS Classification		
Category type	QoS category	Examples
Based on the area of application of the criteria	Generic	Execution Time, dependability, failure rate, price etc.
	Domain-specific	Penalty rate
Based on the knowledge known about the services	Deterministic	Price
	Non-Deterministic	Response time
Based on the way of gathering of the criteria	Subjective	Client's experience
	Objective	Response time, Reliability, Availability
Based on the type of the criteria	Run-Time related	Execution time, throughput, latency
	Management and price related	Regulations, expected/available features, cost, etc.
	Transaction related	Atomicity, Consistency, Isolation, Durability, LRT
	Security related	Access, privacy, data encryption, etc.

Table 2.1: QoS Classifications

related, management and price related, and security related. The run-time category contains features that can be evaluated dynamically, such as scalability, capacity, performance (execution time, latency, and throughput), availability, reliability, error rate, degree of correctness, and error handling. Transaction-related QoS include the characteristics parameters - run-time related, transaction support of the ACID transactions (atomicity, consistency, isolation, and durability) as well as long running transactions (LRT). The third category, management and cost related QoS, consists of criteria related to standards, regulations, expected features/available features, cost as well as updates in the services. The last group represents security related QoS, such as access related features, privacy, and data encryption. Table 2.1 summarizes the QoS classification.

There are different approaches described in the literature for dynamic selection of Web Services that take into account the QoS criteria discussed above. Maximilien and Singh [11] propose a multi-agent based architecture and the use of the Semantic Web in order to select the best service according to the consumers' preferences. In their paper, trust and reputation are taken into account during the decision process. Liu, Ngu, and Zeng [12] consider these features in their proposed approach as well. In addition, other criteria are discussed too - execution price, duration, transactions support, compensation and penalty rate. The authors of [12] suggest an open, fair, and dynamic framework that evaluates the QoS of the available Web Services by using clients' feedback and monitoring.

The solution, suggested by Day and Deters in [13] offers another technique that deals with Web Services selection - a Semantic Web approach that takes into consideration generic criteria - availability, reliability, and execution time of the services, and as in [12], relies on the past experience of different consumers. Padovitz, Krishnaswamy, and Loke [7] talk about availability, reliability, execution time, and service load as criteria that should be taken into consideration when selecting Web Services at run-time.

Ran [16] extends the standard Web Services conceptual model, by adding a new component to the current architecture - a QoS component. This unit contains QoS metrics of the published services, such as scalability, performance (response time, latency, and throughput), reliability, and availability.

Ludwig and Reyhani [17] apply QoS metrics in Grid computing in order to assure dynamic service selection. They take into consideration execution duration, execution price, reputation, reliability, and availability. The services are ranked according to the QoS criteria and the service is chosen by a match-making or a heuristic algorithm.

When dealing with dynamic components like Web Services, it is hard to observe all of their possible features. The researchers focus mostly on some generic criteria, since they can be applied to any service. Availability, reliability, and response time are the most popular ones, as they provide an overview of the services and at the same time they

QoS Criteria	Papers					
	Ran	Ludwig & Reyhani	Day & Deters	Liu, Ngu & Zang	Maximillien & Singh	Padovitz, Krishnaswamy & Loke
Availability	Yes	Yes	Yes	No	No	Yes
Reliability	Yes	Yes	Yes	No	No	Yes
Scalability	Yes	No	Yes	No	No	No
Execution Time	Yes	Yes	Yes	Yes	No	Yes
Trust & Reputation	No	Yes	No	Yes	Yes	No
Network & Machine Load	No	No	Yes	No	No	Yes
Execution price	Yes	Yes	No	Yes	No	No
Transactions	Yes	No	No	Yes	No	No
Compensation rate	No	No	No	Yes	No	No
Penalty rate	No	No	No	Yes	No	No
Security	Yes	No	No	No	No	No
Client's experience / feedback	No	Yes	Yes	Yes	No	No

Table 2.2: Selection criteria for dynamic service selection

can be evaluated relatively easily. A summary of the reviewed literature that uses criteria for dynamic service selection, is shown in table 2.2.

### 2.1 Reasoning mechanism

The selection criteria are one aspect of the decision-making process. Another aspect is the mechanism for Web Service selection. Liu, Ngu, and Zeng [12] propose a framework that evaluates services according to some generic (price, response time, and reputation) and domain specific criteria (business related - transactions support, compensation and penalty rates). The architecture is flexible and allows extension of the QoS parameters according to the characteristics of the system. The service related attributes are given by the providers, evaluated by the consumers via monitoring, or gathered through clients' feedbacks, and are stored in a database. The reasoning mechanism in this approach computes the QoS of according to some generic (price, response time and reputation) and domain specific criteria (business related -transactions support, compensation and penalty rates). The architectures is flexible and allows extension of the QoS parameters according to the characteristics of the Web Services, ranks them, and selects the most appropriate one. The bottleneck of the approach is the dependency on the consumers to give regular feedback about their past experience with the Web Services. An overview of the architecture is shown in figure 2.1.

Day and Deters [13], proposes that it is the client's responsibility to detect the most appropriate service, depending on its configuration and requirements. An overview of the architecture is shown in figure 2.2. It is based on the Semantic Web and uses a service evaluation function and an expert system during the Web Services selection process. However, it does not provide service transparency, since the consumers decide which Web Service should be invoked. In addition, it relies on the clients participation to report experience with the providers. Furthermore, each client requires a reasoning mechanism which augments its complexity. In some cases, the approach can be valuable, because the consumers would be able to choose the best Web Service themselves, according to their needs and preferences, but in other cases the clients cannot be involved in this process. Another approach for Web Services selection is presented by Padovitz, Krishnaswamy, and Loke in [14]. An overview of the architecture is shown in figure 2.3. The client chooses at run-time which service to handle the request. In order to make a decision, service related information must be available to the consumer.

The reasoning approach in [14] as well as the mechanisms provided in [12] and [13] require the clients' participation - to gather run-time information related to the services, to share information about the interactions with the services, to evaluate the QoS of the Web Services, and even to take a decision which service matches their

needs and preferences. These techniques require each consumer to contain an intelligent mechanism which makes it more complex. On another hand, if the clients are responsible for the decision process, it might lead to a situation where multiple consumers have chosen the same provider and service overload has occurred. Furthermore, if the requesters have a local knowledge about the services, it is even possible that the chosen components are not the most appropriate ones. This can happen if the clients collect information only about their interactions with the services and does not have access to all interactions in the system. A multi-agent approach for dynamic service selection that does not increase the consumers' complexity is presented in [11] by Maximilien and Singh. An overview of the architecture is shown in figure 2.4. The authors propose an architecture that consists of interacting agents. Every Web Service has an autonomous agent attached to it, which offers the same interface as the service. This interface allows the consumer and the service agent to communicate. In addition, the agents interact and share information through agencies. The agencies contain data about the interactions between the clients and the services which is used during the Web Services selection process. Furthermore, the reasoning mechanism of the approach is based on ontology and matching algorithm and is used to match semantically clients to services, in a manner transparent to the consumer. The discussed approaches are presented in table 2.3. A review of different selection techniques is provided in the next section.

## 2.2 Selection techniques

A selection technique consists of a set of instructions (steps, rules) that should be followed in order to select a Web Service at run-time, depending on the information that is available to the system. Selection techniques can be divided into several groups, according to their nature as well as to the information that they use about the services. Usually, the approaches are combined together to achieve a more powerful reasoning mechanism. However, in this work the groups of selection techniques are reviewed separately in order to distinguish the principles of the techniques from one another:

- Selection technique that is not based on any information about the services.
- Selection technique that assures system dependability.
- Selection technique that assures better response time.
- Selection technique that assures load balancing.

- Selection techniques that take into account past and current information about the services ([15], [14], [13]):
- Selection technique based on Semantics of services

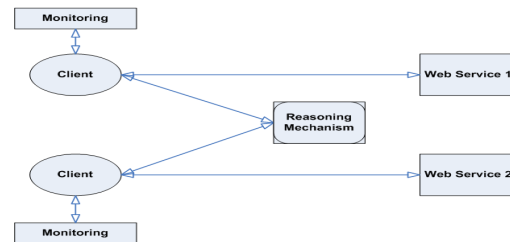


Figure 2.1: Overview of the architecture proposed by Liu, Ngu, and Zeng in [12]

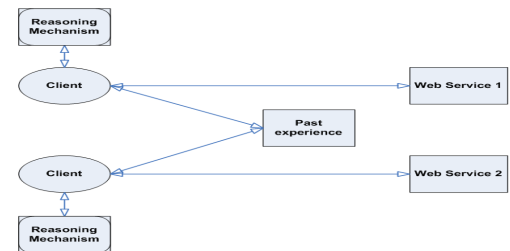


Figure 2.2: Overview of the architecture proposed by Day and Deters in [13]

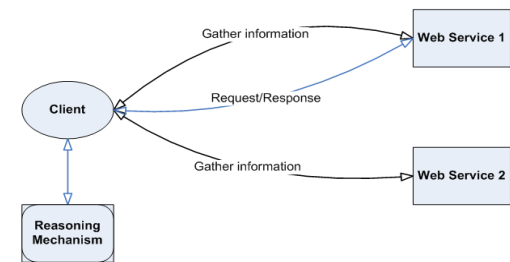


Figure 2.3: Overview of the architecture proposed by Padovitz, Krishnaswamy, and Loke in [14]

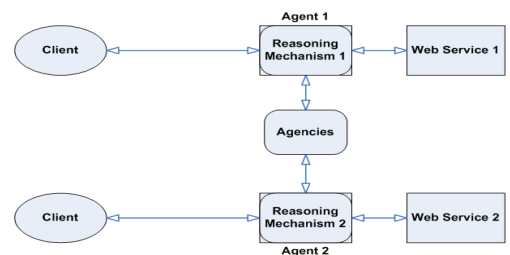


Figure 2.4: Overview of the architecture proposed by Maximilien and Singh in [11]

	Papers			
	Liu, Ngu & Zeng	Day & Deters	Maximilien & Singh	Padovitz, Krishnaswamy & Loke
Dynamic Service Selection	Yes	Yes	Yes	Yes
Selection transparency	No	No	Yes	No
Clients involved in QoS computation	No	Yes	No	Yes
Clients involved in Web Services selection	No	Yes	No	Yes
Clients give feedback	Yes	Yes	No	No
Reasoning mechanism	QoS computation, QoS ranking, DB storage	Semantic Web, Services evaluation function, expert systems, DB storage	Agents, Semantic Web, Matching algorithm, DB storage	RPC / Mobile agents, ranking, DB storage

Table 2.3: Architectures Comparison

### 3. PROBLEM DEFINITION AND PROPOSED ARCHITECTURE FOR WEB SERVICE SELECTION

*Web Service Selection:* Dynamic Web Service selection refers to choosing the available WSs to be invoked so as to realize the functionality of a composite[18]. The purpose of web service (WS) selection is to select optimal web service for a particular task. When dynamic discovery is used in Web Services, it is common that the result of the discovery contains more than one provider. Even for a composite Web Service consisting of many atomic Web Services, the selection issue still needs to be addressed when there are multiple providers available for an atomic service. In order to make a distinction between the services which provide the same functionality, selection criteria should be used. They help evaluate the Web Services within a group and choose the component that matches the needs and the preferences of the consumers, while taking into account the abilities of the providers. Web Services can be ranked by the Quality of Service (QoS) they offer. QoS is a means to enable selection and filter out unqualified providers. QoS can be seen as an aggregated measure of generic criteria such as availability, reliability failure rate, trust and reputation, response time, price, and network load and domain specific features [10].The reasoning mechanism is responsible for the

selection of a Web Service at a particular moment of time. The previous works as already discussed in section 2.2 and as compared in table 2.3, reflects that the clients are not involved in the selection process.

This work proposes as architecture for dynamic and transparent service selection. In this proposed model the clients are also involved in the selection process but the systems complexity is kept hidden from them to enhance the security.

#### 3.1 Proposed Architecture

We propose a technique for dynamic selection of Web Services which will also handle the problem of redundant Web Services. In this work,

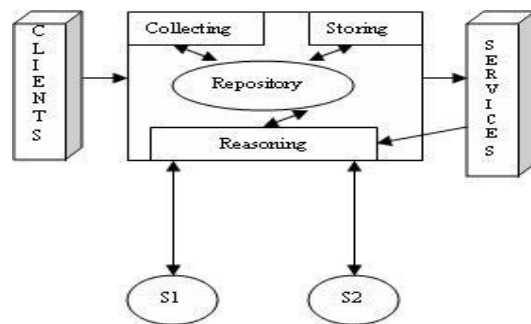
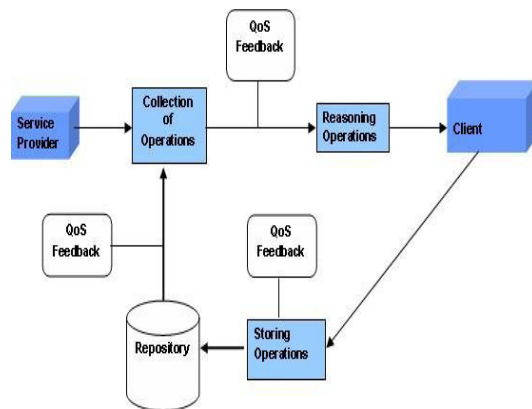


Figure 3.1: Repository based Web Service Selection (Proposed Architecture)

we introduce an architecture with a Web Service repository, as shown in figure 3.1 will act as an independent unit possessing a definite functionality. This repository will be used to redirect the client's request. The proposed architecture will also provide a level of security since it will not be allowed to invoke directly by the clients and also will prevent unauthorized access to the real services. The provision provided in the architecture will also help to hide the systems complexity from the clients.

The repository will perform three functions namely, storing, collecting and reasoning. In storing operation a QoS feedback report is generated by the client and is saved in the repository. The QoS feedback report provides a historical reference for the consumer to assess the provider. Each provider only keeps the feedback information relevant to it. The collecting operation retrieves all necessary data from providers for the reasoning operation. The reasoning operation manages to select the best service provider for the consumer according to the collected data. Consider an example where clients needs a service(S1,S2) as in figure 3.1, it sends a request .The collecting, storing and reasoning mechanism interacts with the web services to find the most appropriate of the services and the results are stored in the repository for future reference. Web Services here interacts with the reasoning mechanism to find out the appropriate services. Once the service is selected, the request is forwarded to it. Web Service selection process is shown in the figure 3.2. Finally, when the result is generated, it is passed to the repository which sends it back to the client.



**Figure 3.2:** Web Service Selection Process

*Algorithm for Web Services Selection:* This algorithm shows the necessary steps to choose a service and get the maximum quality results.

- For finding a service for a specified task, perform a search on service descriptions.

- Arrange all discovered services by their signature parameter and discard all other services.
- Get the desired Service Parameters.
- Collect the services result and order by their utility.
- If no results are found, let the client reconsider the constraints, go to step 2.

This algorithm provides an approach for selecting a specified service. Services, not matching the profile are discarded on the fly. It also helps in taking an alternative services.

#### 4. CONCLUSION

We propose an approach for dynamic service selection and, which has the following advantages in comparison with previous approaches:

- It hides the system's complexity from the clients.
- It provides location and replication transparency of the Web Services
- It provides a transparent service selection from the client's point of view.
- It assures a level of security, since the clients do not have direct access to the Web Services.
- The architecture is flexible, since various reasoning mechanisms can be applied, without modifying the virtual services

In future, other technology that can be applied in the Repository based system is Semantic Web technology. By describing the data in a machine-understandable manner and creating semantics of QoS criteria, the decision-making process would be based on more features as well as their flexible way. The experimentation will be conducted in a reference system, which is a working implementation of the architecture and consists of prototype clients and Service Providers running on multiple machines. The scalability of the reference system is limited to the available hardware resources.

#### 5. REFERENCES

- [1]. IBM Web Services Architecture team. Web Services architecture overview. <http://www-28.ibm.com/developerworks/web/library/w-ovr/?dwzone=web> Last access: 2007-06-18, (7 pages), 2000.
- [2]. M. Papazoglou and D. Georgakopoulos. Service-Oriented Computing. Communications of the ACM, 46(10):25–28, October 2003.

- [3]. M. Papazoglou, P. Traverso, S. Dustdar, F. Leymann, and B. Kramer. Service-Oriented Computing Research Roadmap. <http://drops.dagstuhl.de/opus/volltexte/2006/524/pdf/05462.SWM.Paper.524.pdf> Last access: 2007-06-18, (29 pages), April 2006.
- [4]. Tim Bray, Jean Paoli, C.M. Sperberg-McQueen, Eve Maler, and Fran-cois Yergeau. Extensible markup language (xml) 1.0 third edition) <http://www.w3.org/Protocols/rfc2616/rfc2616.html>.
- [5]. Nilo Mitra. Soap version 1.2 part 0: Primer. <http://www.w3.org/TR/soap12-part0/>.
- [6]. Erik Christensen, Francisco Curbera, Greg Meredith, and Sanjiva Weerawarana. Web services description language (wsdl) 1.1. <http://www.w3.org/TR/wsdl>.
- [7]. UDDI Browser, <http://www.uddibrowser.org>.
- [8]. E. Maximilien and M. Singh. Towards Autonomic Web Services, Trust and Selection. ICSOC'04 pages 212–221, November 2004.
- [9] S. Ran. A Model for Web Services Discovery With QoS. ACM 2003, Volume 4(Issue 1):1–10, March 2003.
- [10]. E. Maximilien and M. Singh. A Framework and Ontology for Dynamic Web Services Selection. IEEE Internet Computing, pages 84–93, September-October 2004.
- [11] E. Maximilien and M. Singh. Toward Autonomic Web Services Trust and Selection. ICSOC'04, pages 212–221, November 2004.
- [12]. Y. Liu, S. Ngu, and L. Zeng. "QoS Computation and Policing in Dynamic Web Service Selection" WWW2004, (8 pages), May 2004.
- [13]. J. Day and R. Deters. Selecting the Best Web Service. CASCON, pages 293–307, October 2004.
- [14]. A. Padovitz, S. Krishnaswamy, and S. Loke. Towards Efficient Selection of Web Services. 2<sup>nd</sup> International Joint Conference on Autonomous Agents and Multi-Agent Systems, (9 pages), July 2003.
- [15]. J. Silva and N. Mendonca. Dynamic Invocation of Replicated Web Services. 10th Brazilian Symposium on Multimedia and the Web 2nd Latin American Web Congress, (8 pages), 2004.
- [16]. S. Ran. A Model for Web Services Discovery With QoS. ACM 2003, Volume 4(Issue 1):1–10, March 2003.
- [17]. S. Ludwig and S. Reyhani. Selection Algorithm for Grid Services based on a Quality of Service Metric. 21st International Symposium on High Performance Computing Systems and Applications (HPCS'07), (7 pages), 2007.
- [18]. San-Yih Hwang, Member, Ee-Peng Lim, Chien-Hsiang Lee, and Cheng-Hung Chen, Dynamic Web Service Selection for Reliable Web Service Composition, IEEE Transactions on Services Computing, Vol. 1, NO. 2, pages 104-116, April-June 2008.