A Model for Web Services Discovery With QoS

SHUPING RAN
CSIRO Mathematical and Information Sciences
GPO Box 664, Canberra, ACT 2601, Australia
Shuping.Ran@csiro.au

Web services technology has generated a lot interest, but its adoption rate has been slow. This paper discusses issues related to this slow take up and argues that quality of services is one of the contributing factors. The paper proposes a new Web services discovery model in which the functional and non-functional requirements (i.e. quality of services) are taken into account for the service discovery. The proposed model should give Web services consumers some confidence about the quality of service of the discovered Web services.

Additional Key Words and Phrases: Web services discovery, quality of services, UDDI, UDDI extension, tModel, model.

1. INTRODUCTION

Web services technology is becoming increasingly popular because of its potential in many areas. It is a new type of components that can be invoked over the Internet. This presents a promising solution for addressing platform interoperability problems faced by system integrators. The flexibility of this new component type also facilitates service composition using existing Web services, promoting component re-use which has been a dream for the software engineering industry. Because of its potential for service composition, agent research community has also explored it for composing agent's behaviors [Buhler and Vidal 2003, Mcilraith et. al. 2001].

Web services technology is now over two years old. Although it has a lot of potential, but the adoption rate has been very slow. According to Gartner research presented at Gartner Symposium ITxpo 2001 [Plammer and Andrews 2001], Web services technology's real take up is by 2005. There are many factors that may contribute to this slow take up, such as perceived lack of security and transaction support [DuWaldt and Trees 2002]. Although there are emerging standards in these areas such as WS-Coordination [BEA, IBM and Microsoft 2002a], WS-Transaction [BEA, IBM and Microsoft 2002b], WS-Security [IBM, Microsoft and Verisign 2002] etc., a coherent picture with full support in all these areas is yet to be seen. Another very important issue is the quality of the Web services [DuWaldt and Trees 2002, Rao 2002, Borck 2001]. At the present time, Universal Description, Discovery and Integration of Web services (UDDI) [OASIS 2002] based look ups for Web services are based on the functional aspects of the desired Web services. Figure 1 presents this publish-find-bind model. Web services technology has yet to address questions such as *how will I know the Web service will meet my performance requirements such as 2 ms response time? Will the Web service be reliable for my mission-critical system's implementation*? Until these questions have been addressed, it is unrealistic to expect that a business would want to search for a Web service based on the expected functional requirements in an UDDI registry and invoke that service without the assurance of knowing the expected quality of service would be met before hand.

To address these problems, this paper proposes a new service discovery model where quality of service is taken as constraints when searching for Web services. This would give some confidence to the Web service consumers about the quality of the service they are about to invoke.

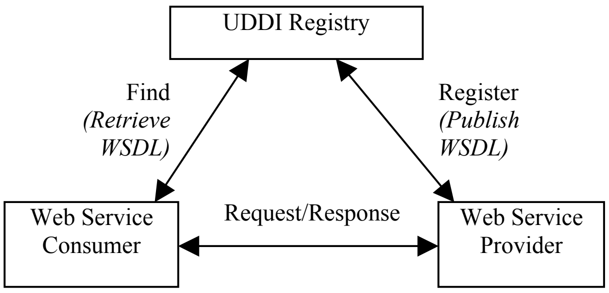

Fig. 1. Current Web services publish-find-bind model.

2. EXTENDING UDDI MODEL

2.1 A New Model

The current proposed Web services publish and discovery model (Figure 1) is largely unregulated based on UDDI registries. 48% of the production UDDI registry (*tModels* tested only) has links that are unusable. These pointers contain missing, broken or inaccurate information [Clarke 2001]. This is one example illustrating the importance of addressing quality of service (QoS) issues. The other shortcoming of the current UDDI model is that it limits the service discovery to functional requirements only. It is foreseeable that there may be more than one Web services available that can meet the functional requirements with different quality of service attributes. Therefore the ability of incorporating quality of service into service discovery process becomes very important. To overcome these shortcomings we propose a new model shown in Figure 2.

The proposed framework is a regulated model that can co-exist with the current de-regulated UDDI registries. The current de-regulated registries can offer services to people to whom the quality of service is not important. The regulated registries based on the model presented here can serve to the applications needing quality of service assurance.

There are four roles in this proposed model: Web service supplier, Web service consumer, Web service QoS certifier, and the new UDDI registry. As before, the Web service provider offers Web service by publishing the service into the registry; the Web service consumer needs the Web service offered by the provider; the new UDDI registry is a repository of registered Web services with lookup facilities; the new certifier's role is to verify service provider's QoS claims described below. The proposed new registry differs from the current UDDI model by having information about the functional description of the Web service as well as its associated quality of service registered in the repository. Lookup could be made by functional description of the desired Web service, with the required quality of service attributes as lookup constraints. The new role in this model is the Web service QoS certifier that does not exist in the original UDDI model. The certifier verifies the claims of quality of service for a Web service before its registration. The details of Web service registration, discovery and invocation are discussed below.

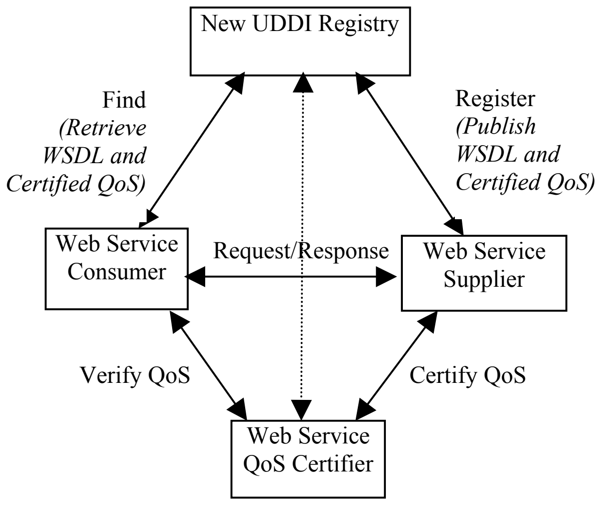

Fig. 2. A new Web services registration and discovery model.

2.2 Web Services Registration Under New Model

In the proposed model, a Web service provider needs to supply information about the company, the functional aspects of the provided service as requested by the current UDDI registry, as well as to supply quality of service information related to the proposed Web service. The claimed quality of service needs to be certified and registered in the repository.

The Web service provider first needs to communicate its QoS claim to the Web service QoS certifier. The certifier checks the claims and either certifies or down grade the claim. The outcome is sent back to the provider with certification identification information. This information is also registered in the certifier's repository identified by a certification Id. The certifier provides a set of Web services for any interested parties to access its repository about QoS claims for verification purposes. After the QoS certification been issued by the certifier, the supplier then registers with the UDDI registry with both functional description of the service and its associated certified quality of service information. The UDDI registry communicates with the certifier to check the existence of the certification. After successful checking, the registry then registers the service in its repository.

2.3 Web Service Discovery and Invocation Under New Model

A consumer of a Web service has certain functional and quality of service requirements, such as "response time not greater than 2 ms with cost less than $100 per invocation". The consumer searches the UDDI registry for a Web service with the required functionality as usual; they can also add constraints to the search operation. One type of constraint is the required quality of service. If there were multiple Web services in the UDDI registry with similar functionalities, then the quality of service requirement would enforce a finer search. The search would return a Web service that offers the required functionality with the desired set of quality of service. If there is no Web service with these qualities, feedback is given to the consumer. The consumer can then reduce their quality of service constraints or considering trade-offs between the desired qualities of service. Once a Web service is found, the WSDL and the certified QoS information is retrieved by the consumer. The consumer can verify the QoS claims with the certifier using the certification Id. Once the consumer is happy with their findings, they can invoke the Web service as per current model.

3. EXTENDING UDDI DATA STRUCTURE

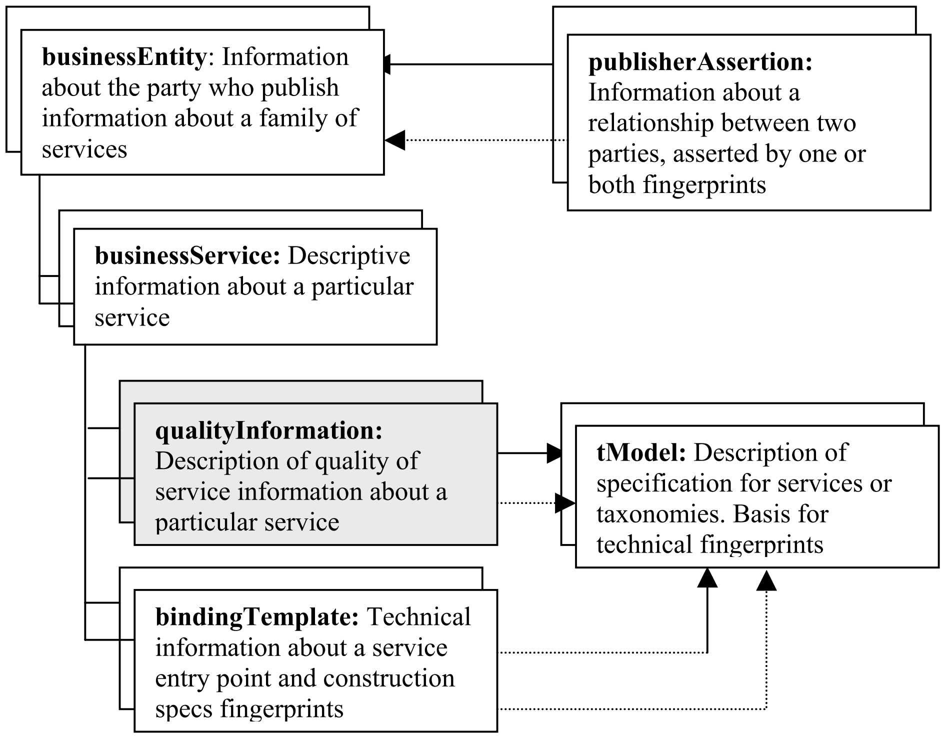
businessEntity: Information about the party who publish information about a family of services
publisherAssertion: Information about a relationship between two parties, asserted by one or both fingerprints
businessService: Descriptive information about a particular service
qualityInformation: Description of quality of service information about a particular service
tModel: Description of specification for services or taxonomies. Basis for technical fingerprints
bindingTemplate: Technical information about a service entry point and construction specs fingerprints

Fig. 3. UDDI data structure types including the new proposed quality information data type.

The information that makes up an existing UDDI registration consists of five data structure types [UDDI Committee 2002]: businessEntity, businessService, bindingTemplate, publisherAssertion and tModel. These are presented as un-shaded boxes in Figure 3. These five types make up the complete amount of information provided within the current UDDI service description framework. Each of these XML structures contains a number of data fields that serve either a business or technical descriptive purpose. [UDDI Committee 2002] explains each of these structures and the meaning and placement of each field. These structures are defined in the UDDI Version 2.0 API schema. The schema defines approximately 25 requests and 15 responses, each of which contain these structures, references to these structures, or summary versions of these structures.

In order to realize the proposed extension to the UDDI model discussed in Section 2, we propose to add a new data structure type. It is presented as a shaded box in Figure 3. This data structure type represents description of quality of service information about a particular service. Different categories of quality of service information can be provided under this *qualityInformation* data structure, such as availability, reliability [Gunther 1998] etc. Section 5 discusses the potential categories of QoS in more detail. This proposed data structure is under the *businessService* data structure type, in addition to *bindingTemplate* data structure type, which provides binding information for a particular service.

Like *bindingTemplate*, this new data structure also refers to *tModels* defined in the UDDI registry. Unlike *bindingTemplate* refers to *tModels* as reference to interface specifications of services, *qualityInformation* refers to *tModels* as references to quality of service taxonomies which also need to be defined in the extended UDDI registry. These taxonomies define the new terminologies or concepts about the proposed QoS information, which do not exist in the existing UDDI registries. Figure 4 shows an example of one of these proposed tModels. The tModel defines the term *qualityInformation* in this case.

```
<tModel tModelKey="uuid:0e727db0-3e14-11d5-98bf-002035229c64">
    <name>uudi-org:qualityInformation</name>
    <description xml:lang="en">Quality of Service Information</description>
    <overviewDoc>
        <description xml:lang="en"></description>
        <overviewURL> http://www.uddi.org/specification.html
        </overviewURL>
     </overviewDoc>
     <categoryBag>
         <keyedReference
            keyName="uddi-org:types" keyValue="categorization"
            tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"/>
         <keyedReference
            keyName="uddi-org:types" keyValue="checked"
            tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"/>
        <keyedReference
            keyName="uddi-org:types" keyValue="specification"
            tModelKey="uuid:c1acf26d-9672-4404-9d70 39b756e62ab4"/>
     </categoryBag>
</tModel>
```

Fig. 4. tModel for quality of service information (*qualityInformation).*

```
<?xml version="1.0" encoding="UTF-8" ?>
   <envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
      <body>
         <find_service businessKey="*" generic="1.0"
             xmlns="urn:uddi-org:api" maxRows="100">
             <findQualifiers></findQualifiers>
            <name>Stock quote</name>
            <qualityInformation>
              <availability> 0.9 </availability>
            </qualityInformation>
         </find_service>
      </body>
   </envelope>
```

Fig. 5. SOAP request for service discovery

4. PUTTING IT TOGETHER – A SERVICE DISCOVERY EXAMPLE

Figure 5 shows a service discovery request example using SOAP, where the required service is related to *Stock Quote* with a desired QoS attribute *availability* at least of 0.9 (probability of the service available 90%, see next section). Figure 6 shows the corresponding SOAP response. In this case, there are two services satisfying the required quality of service: *Stock Quote* and *Stock Quotes* from two different service suppliers identified by *businessKey* fields *b42b5fef-85df-4fbf-b468-62a356089ea8* and *b6cb1cf0-3aaf-11d5-80dc-002035229c64* respectively. The Web service requester can then decide to choose one from this list of service providers after receiving the response.

```
<?xml version="1.0" encoding="UTF-8" ?>
<envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
   <body>
         <serviceList  generic="1.0" xmlns="urn:uddi-org:api"
              operator="www.ibm.com/services/uddi" truncated="false">
              <serviceInfos>
                 <serviceInfo
                     serviceKey="9021cb6e-e8c9-4fe3-9ea8-3c99b1fa8bf3"
                     businessKey="b42b5fef-85df-4fbf-b468-62a356089ea8">
                     <name>Stock Quote</name>
                     <qualityInformation>
                         <availability> 0.99 </availability>
                     </qualityInformation>
                 </serviceInfo>
                 <serviceInfo
                     serviceKey="74154900-f0b0-11d5-bca4-002035229c64"
                     businessKey="b6cb1cf0-3aaf-11d5-80dc-002035229c64">
                     <name>Stock Quotes</name>
                     <qualityInformation>
                        <availability> 0.91 </availability>
                     </qualityInformation>
                 </serviceInfo>
               </serviceInfos>
          </serviceList>
   </body>
</envelope>
```

Fig. 6. SOAP response to the SOAP request for service discovery.

5. QUALITY OF SERVICE DESCRIPTION

Quality of Service research has been an active research area for several domains. The term "quality of service" has been used for expressing non-functional requirements for different areas such as network research community [Cruz 1995, Salamatian and Fdida 2001] and in real time issues [Clark, Shenker and Zhang 1992]. There is some research effort in defining QoS in distributed systems. Their interests are primarily on how to express the QoS for a system, and how these requirements are propagated to the resource manager to fulfill the QoS requirements [Tien, Villin and Bac 2000]. [Stephanie et. al. 1997] presents a layered model for representing QoS for telecommunication applications. It presents service quality function, QoS schema mapping and price-QoS trade-off.

[Frølund and Koistinen 1998] presents a QoS specification language. They advocate its use for designing distributed object system, in conjunction with the functional design. [Sabata et. al. 1997] categorizes the QoS from different viewpoints: application, system and resource. It specifies QoS in terms of metrics and policy. All these research categorizes and define QoS from their perspective with some overlap between them. There is not great consensus about a set of QoS important to distributed systems. There is even less research done on the QoS for service-oriented architecture, although QoS is an important aspect as discussed in the introduction Section.

Because Web services can be provided by third parties and invoked dynamically over the Internet, their QoS can vary greatly. Therefore it is important to have a framework capturing the QoS provided by the supplier and for the QoS required by the customer, and ultimately the match between the two when discovering the Web service best match the required QoS.

The international quality standard ISO 8402 (part of the ISO 9000 (ISO9000 2002)) describes quality as *"the totality of features and characteristics of a product or service that bear on its ability to satisfy stated or implied needs*." We define quality of service as a set of non-functional attributes that may impact the quality of the service offered by a Web service in the context of this paper.

There are many aspects of QoS important to Web services. We are starting to organize them into QoS categories. Each category needs to have a set of quantifiable parameters or measurements. Further research is needed in this area. For illustration purposes, the categories are defined here. They described briefly, followed by a question as an example to show what type of questions the particular QoS can address. To facilitate the description, the categories are grouped into different types, i.e. QoS related to runtime, transaction support, configuration management and cost and security.

5.1 Runtime Related QoS

Scalability – The capacity of increasing the computing capacity of service provider's computer system and system's ability to process more operations or transactions in a given period. It is related to performance.
Q: Will the system scale up to handle X transactions per second? This is closely related to throughput and performance.

Capacity – Limit of concurrent requests for guaranteed performance.
Q: How many concurrent connections does the service support?

Performance – a measure of the speed in completing a service request. It is measured by:

> *Response time* – the guaranteed max (or average or min) time required to complete a service request (related to capacity [Gunther 1998]).
> *Latency* – Time taken between the service request arrives and the request is being serviced.
> Q: What is the average delay on servicing a request?
> *Throughput* **–** The number of completed service requests over a time period. Throughput is related to latency/capacity.

Reliability **–** The ability of a service to perform its required functions under stated conditions for a specified period of time [Institute Of Electrical And Electronics Engineers 1990]. It can be measured by: Mean time between failure (MTBF), Mean Time to Failure (MTF), and Mean Time To Transition (MTTT). It is closely related to availability [Gunther 1998].

Availability **–** It is the probability system is up and related to reliability. It can be measured as [Gunther 1998]:

$$A = \frac{\langle upTime \rangle}{\langle totalTime \rangle} = \frac{\langle upTime \rangle}{(\langle upTime \rangle + \langle downTime \rangle)}$$

Where:

<*upTime*> is the total time the system has been up during the measurement period.

<*downTime*> is the total time the system has been down during the measurement period.

<*totalTime*> is the total measurement time, is the sum of <*upTime*> and <*downTime*>.

Q: What is the chance for the service is available when I invoke it?

Robustness/ Flexibility **–** It is the degree to which a service can function correctly in the presence of invalid, incomplete or conflicting inputs.

Q: Will the service still work if incomplete parameters are provided to the service request invocation?

Exception handling – Since it is not possible for the service designer to specify all the possible outcomes and alternatives (especially with various special cases and unanticipated possibilities), exceptions can be expected. Exception handling is how the service handles these exceptions. It can be in a brutal or a graceful way.

Q: How will the service still work correctly if I give less number of parameters than it requires?

Accuracy – Defines the error rate produced by the service.

Q**:** How many errors does the service produce over a period of time?

5.2 Transaction Support Related QoS

Integrity – Transactions can be grouped into a unit in order to guarantee the integrity of the data operated on by these transactions. The unit can either be successful where all transactions in the unit "commit" or all "roll back" to their original state in case of a transaction failure. This is described by the ACID properties: *Atomicity* (executes entirely or not at all), *consistency* (maintains the integrity of the data), *isolation* (individual transactions run as if no other transactions are present) and *durability* (the results are persistent)).

A two-phase commit capability is the mechanism to guarantee the ACID properties for distributed transactions running over tightly coupled systems as if they were a single transaction. It is more difficult in the Web services environment, as the transactions may involve more than one business partner with the possibility of transactions spanning over long time (hours or days) – Long Running Transactions (LRT). The transaction integrity is still described by ACID properties, although it is a much harder to achieve in this case. It may require different mechanisms [Peryret 2002].

5.3 Configuration Management and Cost Related QoS

Regulatory – It is a measure of how well the service is aligned with regulations.

Q: How aligned is the service with appropriate regulations?

Supported Standard – A measure of whether the service complies with standards (e.g. industry specific standards). This can affect the portability of the service and

interoperability of the service with others. One example is ISO 8583, which is a standard for creating and reading financial transaction messages including Point of Sale (POS) transactions [ISO].
Q: How much does the service adhere to applicable standards? Or what standards does the service comply?

Stability/change cycle – A measure of the frequency of change related to the service in terms of its interface and/or implementation.
Q: How stable is the service, how often it changes (interface and implementation)?
Guaranteed messaging requirements – does it ensure the order and persistence of the messages?

Cost – It is a measure of the cost involved in requesting the service.
Q: What is the cost based on (per request or per volume of data)?

Completeness – A measure of the difference between the specified set of features and the implemented set of features.
Q: How many of the specified features are currently available?

5.4 Security Related QoS

It measures of the trustworthiness and mechanisms security implemented.
Authentication – How does the service authenticate principals (users or other services) who can access service and data?
Authorization – How does the service authorize principals so that only them can access the protected services?
Confidentiality – How does the service treat the data, so that only authorized principals can access or modify the data?
Accountability – Can the supplier be hold accountable for their services?
Traceability and Auditability – Is it possible to trace the history of a service when a request was serviced.
Data encryption – How does the service encrypt data?
Non-Repudiation – A principal cannot deny requesting a service or data after the fact.
Q: How does the service provider ensure these security requirements?

6 CONCLUSIONS AND DISCUSSION

This paper discussed issues related to Web services technology's slow take up and proposed that quality of services is one of the issues contributing factors. The paper proposed a new Web services discovery model in which the functional and non-functional requirements (i.e. quality of service) are to be taken into account for the service discovery. A new role is introduced into this framework – the Certifier(s). They verify the QoS claims from the Web service suppliers. Their role is very similar to rating agencies in other domains such as the financial sector, service industry etc. The paper also proposed an extension to UDDI's data structure types that could be used for implementing the proposed extended UDDI model.

In order for the proposed framework to be realized, we need to establish a set of metrics to quantify each QoS category proposed here and their associated models for their representation. Further research is needed in establishing the matching algorithms between the desired and supplied QoS. It is anticipated that a consumer of a service may not need all the QoS categories. The matching algorithms need to take this into account. To fully exploit the potential of the proposed framework, incorporating the semantic modeling of the QoS categories is necessary. The details for the Certifiers also need to be explored further.

REFERENCES

APICELLA, A. 2002. Making application ends meet, *InfoWorld*. http://www.infoworld.com/article/02/02/22/020225fecost_1.html?Template=/storypages/ctozone_story.html

BEA, IBM AND MICROSOFT. 2002a. Web Services Coordination (WS-Coordination), http://www-106.ibm.com/developerworks/library/ws-coor/

BEA, IBM AND MICROSOFT. 2002b. Web Services Transaction (WS-Transaction), http://www-106.ibm.com/developerworks/webservices/library/ws-transpec/?dwzone=webservices

BORCK, J. R. 2001. Shoring up Web services, http://archive.infoworld.com/articles/fe/xml/01/12/03/011203fewebnets.xml

BUHLER, P. AND VIDAL, J. M.2003. Semantic web services as agent behaviours, *Agentcities: Challenges in Open Agent Environments*, pages 25-31. Springer-Verlag,

FRØLUND, S. AND KOISTINEN, J. 1998. Quality-of-service specification in Distributed object systems, *Distributed System Engineering 5*: 179–202.

IBM, MICROSOFT AND VERISIGN. 2002. Web Services Security (WS-Security) Version 1.0, http://www-106.ibm.com/developerworks/webservices/library/ws-secure/

CLARK, M. 2001. UDDI weather report. http://www.webservicesarchitect.com/content/articles/clark04.asp

CRUZ, R. L. 1995. Quality of service guarantees in virtual circuit switched networks. *IEEE J. Select. Areas Commun*. 13(6): 1048-1056.

CLARK, D. D., SHENKER, S. AND ZHANG, L. 1992. Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism. *SIGCOMM 19*92: 14-26.

DUWALDT AND TREES. 2002. *Web Services A Technical Introduction*, DEITEL™ Web Services Publishing.

GUNTHER, N. J. 1998. *The Practical Performance Analyst*, published by McGraw-Hill.

ISO. http://www.iso.ch/iso/en/ISOOnline.frontpage

MCILRAITH, S. A., SON, T. C., AND ZENG, H. 2001**.** Semantic Web Services, *IEEE Intelligent Systems, Special Issue on the Semantic Web*, Volume 16, No. 2, 46-53.

OASIS. 2002. Universal Description, Discovery and Integration of Web Services (UDDI) Version 2.0, http://www.uddi.org/

PERYRET, H. 2002. Mission-Critical Web Services: Plan for Long Running Transactions, *IDEABYTE*. http://www.hpmiddleware.com/downloads/pdf/giga_report.pdf

PLUMMER, D., AND ANDREWS, W. 2001. The Hype Is Right: Web Services Will Deliver Immediate Benefits. http://www3.gartner.com/DisplayDocument?id=344028&ref=g_search

RAO, A. 2002. Web Services Unleashed, *garage insight* vol2. http://www.garage.com/newsletter/index.shtml

SABATA, B., CHATTERJEE, S., DAVIS, M., SYDIR, J. J. AND LAWRENCE, T. F. 1997. Taxomomy of QoS Specifications, *WORDS'97*, 100-107

INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. 1990. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York, NY.

SALAMATIAN, K. AND FDIDA, S. 2001. Measurement based modeling of quality of service in the Internet: a methodological approach, *IWDC 2001*:158-174.

STEPHANIE, T. H., et. al. 1997. Service Quality in TINA- Quality of Service Trading in Open Network Architecture, *Proceedings of the 1st International Enterprise Distributed Object Computing Conference (EDOC'97):* 322-333.

TIEN, L. D., VILLIN, O., AND BAC, C. 2000. CORBA Application Tailored Manager for Quality of Service Support, *Third IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, pp. 52-59.

UDDI COMMITTEE. 2002. UDDI Version 2.03 Data Structure Reference, http://www.uddi.org/pubs/DataStructure_v2.htm.