QoS-aware cross-layer communication for Mobile Web services with the WS-QoS framework

M. Tian, A. Gramm, H. Ritter, J. Schiller, and T. Voigt*
Freie Universität Berlin, Institut für Informatik
{tian, gramm, hritter, schiller}@inf.fu-berlin.de
* Swedish Institute of Computer Science, thiemo@sics.se

Abstract: QoS issues will play an important role for the success of Web services. With the increasing number of mobile devices consuming Web services, specific QoS mechanisms are required for the efficient use of Web services. We introduce our Web service QoS framework that is designed for QoS-aware service specification, discovery, selection, and invocation of Web services. Applying the framework enables cross-layer communication in order to achieve cross-layer QoS differentiation. We present an architecture based on this framework supporting mobile and wireless Web service clients.

1 Introduction

Web Services are becoming more and more popular these days and more and more businesses are planning to build their future solutions on the Web service technology. With its tremendous success in business, quality of service (QoS) issues will play an increasing role for Web service providers. With the ubiquity of mobile devices, such as Smartphones and PDAs, it is easy to imagine that in the future clients using mobile devices will generate a large percentage of all Web service requests.

In contrast to traditional web interaction, Web services incorporate additional and non-negligible overhead due to the usage of XML. Since mobile devices are resource-constrained in terms of CPU, memory, and battery-life, they need specific QoS mechanisms to efficiently process and transmit Web services.

In this paper we present our Web service QoS (WS-QoS) framework [Ti03a], which has been designed to provide a solution to QoS-aware specification, selection, publication, and invocation of mobile Web services, thus to achieve an overall performance improvement.

The rest of the paper is outlined as follows: After discussing related work, we present our solution in Section 3. We conclude and discuss some future work in Section 4.

2 Related Work

Distinguishing QoS will influence strategies for selecting Web services as building blocks to compose more sophisticated business applications. Therefore, features such as security, message reliability and transactional QoS are issues in recent business process management standards, above all the Business Process Execution Language for Web Services (BPEL4WS) [IMB02]. High level QoS support is also being addressed in the work on the WS-Integration specifications.

Several approaches have been proposed to implement standardized QoS specification for Web Services. Both IBM's Web Service Level Agreement (WSLA) framework and the

Web Service Management Language (WSML) applied in HP's Open View Internet Services define XML schemas to specify individually negotiated customized service level agreements (SLAs). A detailed discussion of six approaches towards QoS specification for Web services is given in [Ti03c].

3 The WS-QoS framework

A QoS-aware Web service communication process consists of three phases from the client's point of view. The first one is the specification of QoS requirements. The second one is a QoS-aware service discovery and selection. The specified QoS is performed at service invocation in the third phase. Our WS-QoS framework supports all three phases.

3.1 QoS specification with WS-QoS

We developed the WS-QoS framework with the following motivations:
(1) design an architecture that allows both service clients and service providers to specify requests and offers with QoS properties and QoS classes,
(2) enable an efficient service offer discovery and selection in order to accelerate the overall lookup process for clients,
(3) provide a flexible way for service providers to publish and update their service offers with different QoS aspects as well as
(4) support QoS-aware service invocation and response, including mapping of QoS requirements according to the transport network onto the actual underlying QoS-aware network technology (e.g. UMTS, DiffServ) at run-time, thus to achieve an overall performance gain.
By applying the WS-QoS XML schema service providers can augment their Web service offers with various QoS aspects while clients can define their requirements related to QoS parameters. In an element of the type `QoSInfo`, shown in Figure 1, QoS parameters such as processing time, request rate, response time, availability, reliability, security protocols, transaction, price, and custom defined parameters are declared for the QoS support in the Web service layer by clients and servers. Standard and custom defined parameters such as delay, bandwidth, jitter, and packet loss are defined for the network layer QoS support by both parties. QoS parameters are automatically compared during the service discovery and selection phase.

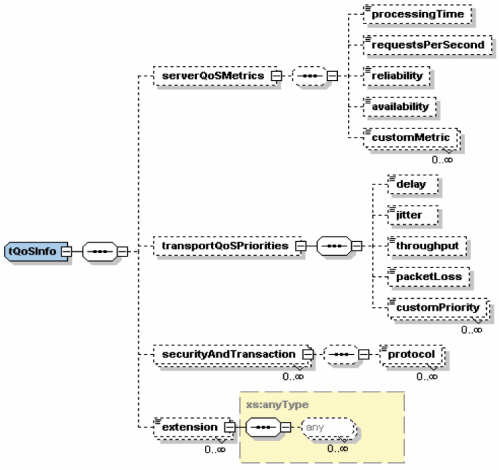

Figure 1. QosInfo element of the WS-QoS XML schema [Ti03a]

3.2 QoS-aware service discovery and selection with WS-QoS

In order to enable an efficient and accelerated QoS-aware service discovery and

selection, we extended the standard Web service interaction model [Kr01] with a Web service offer broker (WSB). The WSB accelerates the client lookup process for services. That means a Web service client will contact the WSB for looking up a service instead of doing this with a UDDI registry. The WSB holds up-to-date information on offers currently available for a group of services which have been requested in recent time. Offers are grouped by the interface (tModel) that the services implement. The first time services for an interface are requested, one or more UDDI registries associated with the WSB are inquired. The WSDL files for these services are then checked for WS-QoS extensions and available offers are built. From then on this newly created offer list is consulted to find the best match for clients and their requirements.

3.3 Cross-layer communication with WS-QoS

In the WS-QoS framework a client defines different QoS aspects/parameters according to different layers such as delay, jitter in the transport layer, compression and decompression algorithms of SOAP content in the SOAP layer, response time, and availability of the Web (service) server in the Web service/application layer. These definitions take place in the Web service layer. The definitions are then interpreted and performed by different components dedicated to each layer.
Figure 2 depicts the WS-QoS architecture during the third phase of a mobile Web service communication process. We assume that the radio bearer supports both QoS and QoS classes such as UMTS and GPRS and that the wired network supports the DiffServ technology, which is, for example, applied by Telecom Italia.

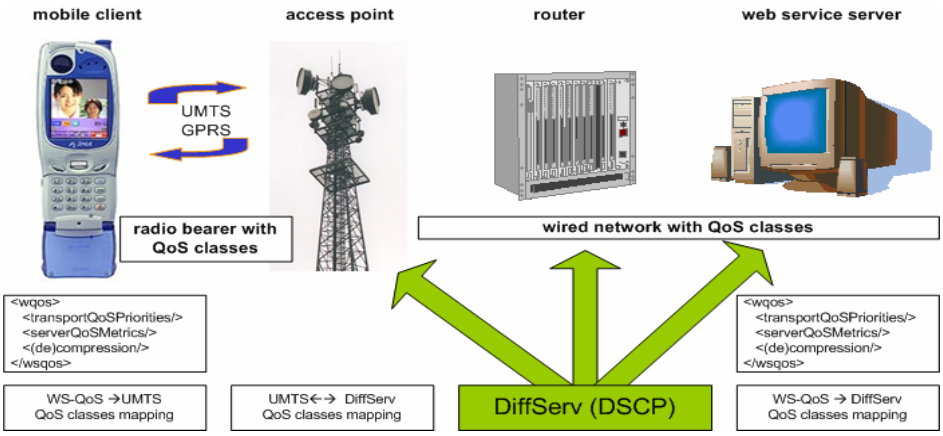

Figure 2. QoS-aware mobile Web service invocation

3.3.1 QoS differentiation on the transport layer

QoS requirements specified by the clients are placed in the WS-QoS SOAP header. Since the SOAP header is part of the SOAP messages, it can be evaluated by every participating component along the whole communication path to allow for a cross-layer QoS differentiation. Since concrete QoS metrics differ for specific network technologies, we have decided to define priorities for QoS parameters according to the transport layer. The priorities are then evaluated by a network proxy specific to a technology, which maps the specified requirements to a corresponding traffic class. We have implemented

such a proxy for DiffServ networks.
In the scenario shown in Figure 2, a QoS-proxy running on the mobile client translates the QoS requirements according to the transport QoS priorities to the corresponding UMTS QoS class and performs signaling with the UMTS system. Since both DiffServ and UMTS support QoS classes, the access point (AP) can now map the UMTS QoS class to a corresponding DiffServ class (DiffServ code point, DSCP) [MNV02]. This task is performed without any knowledge of the WS-QoS framework. Optionally, if the AP would support WS-QoS, it could map the client's requirement to a corresponding DSCP by evaluating the QoS information in the WS-QoS SOAP header.
The intermediate DiffServ-enabled routers treat the traffic depending on the DSCP. Upon receiving the client's request, the server processes the response. When the server sends the response, it will put the client's QoS requirements into the SOAP header again. A server side QoS-proxy will then evaluate the QoS information and mark the DSCP in each IP packets accordingly. The intermediate routers will treat the IP packets according to the DSCP. The AP will map the DiffServ class to a corresponding UMTS class.

3.3.2 Adaptive server performance levels

The `serverQoSMetrics` element of a WS-QoS definition (Figure 1) specifies server performance in terms of processing time, throughput, availability and reliability. Clearly, these parameters are interdependent. A short response time may allow higher throughput and high throughput will provide for high availability. A service offer defines a distinct level of service performance. Request differentiation may take the actual client requirements into account. Yet, a more general approach to differentiate server performance based on the selected offer will be more efficient in terms of scalability.
Request differentiation can take place on various levels. In the current implementation, we consider request differentiation on the application level. Response times are influenced by setting the priority of the thread processing the request according to the clients' requirements. However, different approaches such as load balancing and service differentiation in web servers [Vo01] could be applied to improve server performance.

3.3.3 Message load reduction through adaptive compression

Although mobile devices are resource-constrained, the capability of mobile hardware in terms of CPU power and memory is increasing rapidly. But the improvement and increase of the battery life-time and the data rate for wireless transmission are still challenging issues in active research. Therefore, considering both aspects in mobile computing is essential.
Compression and decompression on mobile devices need not be performed by the same algorithm. Energy consumption can be reduced up to 30% by choosing the lowest-energy compressor and decompressor on a mobile device. Furthermore, wireless transmission of a bit can require 1000 times more energy than a single 32-bit computation [BA03].
To signal what compression is to be used we extend the securityAndTransaction node of the tQoSInfo element, which is defined in the WS-QoS XML schema, with two sub nodes compression and decompression. The servers announce which (de)compression algorithms they support. The clients define which compression algorithm a server has to use to compress responses.

Compression also decreases server performance due to the additional CPU time required. Our measurements in [Ti03b] show that the throughput of a heavily loaded server can decrease substantially when it is required to compress Web service responses. At the same time the response times experienced by the clients increase. In order to protect the server from overloading, we proposed a simple scheme that allows clients to specify whether they want to receive data compressed when requesting a Web service.

4 Conclusions and future work

The value of this paper is not merely to show that WS-QoS is a suitable solution to QoS-aware mobile Web service communication, but rather to stress the fact that the various QoS aspects of distinct communication layers participating in different communication phases should always be considered as parts of an integrated solution. Broader research will be necessary in order to enable different layers to communicate with each other. Higher layers should be prepared to consume QoS provided by lower layers and lower layers should actively provide QoS according to the requirements of the upper ones. In other words, the cooperation and communication patterns of different layers should be mapped to each other carefully. This requires components with some degree of intelligence. Applying the WS-QoS framework enables cross-layer communication in order to achieve cross-layer QoS differentiation. Different parts of the solution have been implemented and performance measurements have been conducted [Ti03b], [Gr03], [Na03]. The results prove the feasibility of the proposed solution. We are now building a testbed to conduct performance measurements of the complete architecture.

5 References

[BA03] K. Barr and K. Asanovic, Massachusetts Institute of Technology, Energy Aware Lossless Data Compression, USENIX MobiSys, 2003.

[Gr03] A. Gramm, QoS support for Web services, diploma thesis, FU Berlin, 2003.

[IMB02] IBM, Microsoft, Bea, Business Process Execution Language For Web Services, 2002.

[Kr01] H. Kreger, Web Service Conceptual Architecture WSCA 1.0, 2001.

[MNV02] S.I. Maniatis, E.G. Nikolouzou, I.S Venieris, QoS issues in the converged 3G wireless and wired networks, IEEE, Communications Magazine, Volume 40, Issue 8, 2002

[Na03] M. Nabulsi, A concept for QoS aware Web services, diploma thesis, FU Berlin, 2003.

[Ti03a] M. Tian, A. Gramm, T. Naumowicz, H. Ritter, J. Schiller. A Concept for QoS Integration in Web Services, IEEE Computer Society 1st Web Services Quality Workshop (WQW 2003) ISBN: 0769521037, Rome, Italy, 2003.

[Ti03b] M. Tian, T. Voigt, T. Naumowicz, H. Ritter, and J. Schiller, Performance Considerations for Mobile Web Services, IEEE Communication Society Workshop on Applications and Services in wireless Networks (ASWN 2003), Bern, Switzerland, 2003.

[Ti03c] M. Tian, A. Gramm, H. Ritter, J. Schiller. A Survey of current Approaches towards Specification and Management of Quality of Service for Web Services. To be published in PIK, Sonderthemenheft "Web services", 2004.

[Vo01] T. Voigt, R. Tewari, D. Freimuth and A. Mehra. Kernel Mechanisms for Service Differentiation in Overloaded Web Servers, Proceedings of Usenix Annual Technical Conference, pages 189 – 202, Boston, MA, USA, June 2001.