1 Introduction

Web services are software components that can be accessed over the Internet using popular web mechanisms and protocols such as the hypertext transfer protocol (HTTP, refer to the list of abbreviations in the Appendix). Public interfaces of Web services are defined and described using extensible markup language (XML) based definitions. The emerging Web service technology has been experiencing an enormous attention and dissemination both in the academic and the industrial world since the publication of the first Web service specifications in 2000. The main reasons for the immense interest are:

The standardization of the Web service technology is driven by leading industrial companies and organizations. Competing companies work together in order to ensure the interoperability of Web service standards. The specifications around Web services are free of royalty. Both software vendors and users need not pay any license fee for their applications based on Web service specifications.

The Web service technology is fully based on XML and Internet technologies such as HTTP, TCP/IP. Therefore, Web services are independent of hardware, programming, software and operating platforms. This allows for a great flexibility for developers and software vendors when developing either Web service infrastructures or Web service applications.

One major business area for Web services is the enterprise application integration. Many companies have different IT infrastructures including

hardware, operating systems, middleware, databases, application servers, and applications. In the past, companies had to spend a lot of time and money to make different IT systems work together, e.g. different wrappers or converters had to be implemented in order to make two different systems interoperable, which is an error-prone and time-consuming task. The effort increased rapidly when different incompatible systems had to cooperate with each other.

With the introduction of Web services, only one software bus has to be created. Only one Web service wrapper around each existing system has to be implemented. All the different systems can now communicate via Web services, resulting in a rapid and efficient development.

Other examples of Web services range from simple requests such as stock quotes or user authentication to more complex tasks such as comparing and purchasing items over the Internet. Famous representatives are Amazon, eBay, and Google Web services [1,2,3]. The Amazon Web service project allows, for example, developers to use the Amazon product database for their own software projects. E.g. one can send the international standard book number (ISBN) of a book to the Amazon Web service, then Amazon returns information about the requested book such as title, author names, review, images of the cover, and price to the service requestor.

Figure 1. A mobile Web service scenario

Figure 1 depicts a mobile Web service scenario, which is an award winning solution [4] designed and implemented at the department of Computer Science, Freie Universität Berlin. The solution called “Smart Shopping” allows consumers to obtain product information including prices. When people go shopping, they do not really know if an offered discount is really a good offer. Smart Shopper helps consumers to check the prices instantly. It works as follows:

1. The consumer uses her mobile device such as a smart phone either to scan the Universal Product Code (UPC) of the product or to enter the product name and description.

2. The Smart Shopper sends the information to the Smart Price Web services via wireless network such GPRS.

3. The Smart Price Web services ask other online Web services such as Amazon, Barnes&Nobbles or eBay for product information and their prices for the product.

4. The Smart Price Web services prepare the results received from various Web services according to the user's profile and return the final result to the consumer.

5. Now, the consumer can decide whether to buy the product in the current department store or online. The Smart Price Web services can also suggest the consumer to go to another department store in the same shopping area when the price there is lower. In this case, a Navigation Web service could be applied in order to route the consumer to that department store.

The increasing industrial and academic involvement in the still emerging Web service technology clearly shows the potential of Web services to become one of the pillars of the software industry. Competing Web services that implement same or similar functionalities will be available on the market, e.g. search engines. Service providers will strive to gain the favor of customers. As offered functionalities are similar, the quality of offered services will be decisive for the success of the service providers. While service offers with no guarantees on throughput, response time, security, availability, reliability, etc. are accepted in some simple cases, most likely this will not be acceptable when a Web service becomes an important part of an application composed of various Web services [5].

Considering non-functional properties of a service, referred to as quality of service (QoS), is essential for the success of Web services. Both service clients and providers need a mechanism to specify their requirements and offers, which can be matched in an easy way. Clients seek to experience a good service performance, e.g. low waiting time, high reliability, and availability to successfully use services. On the other hand, when it comes to e-business, service providers need to formulate QoS-aware offers in order to gain the highest possible profit from their business. Examples are high throughput guarantees and low response time through dynamic capacity allocation, resource allocation, and load balancing in order to serve a high number of clients with assured QoS. Moreover, crucial transactions such as payments should experience prioritized execution realized by transaction differentiation. Service providers will strive to find an optimal relation between user satisfaction and system utilization.

Since Web service providers will be competing for customers by offering the same or similar services, the efficient lookup and selection of services will be an integral part of Web service communication. Customers may want to decide at runtime which Web service provider they use. The decision for that will be based on the QoS offered by the service providers at the time of the service usage.

During the service invocation, it is important that all layers in terms of the Internet Model and all domains participating in Web service communication support QoS. The involved layers are e.g. application, Web service, and network layers. The participating domains are e.g. clients, clients' devices, networks, routers, access points, and servers. An overall QoS guarantee can be given when all these components actively support QoS.

As the Smart Shopper application implies, it is easy to imagine that in the future clients using mobile devices will generate a large percentage of all Web service requests. Although the computing power of handheld devices is increasing rapidly, the data rate over the air and the battery life time are still challenging issues of active research.

With the WS-QoS framework, an architecture targeting the new challenging issues regarding Web services and mobile Web services is introduced. We propose that only an overall QoS consideration including QoS specification, QoS-aware service lookup, and QoS-aware service invocation can guarantee and increase the total performance of Web service communication.

1.1 Targeting issues of this thesis

As service providers will be competing for customers by offering similar or same services, the QoS of service offers will play a tremendous role in their success. Traditionally, QoS is associated with network parameters such as bandwidth, packet loss rate, and jitter. However, QoS in the realm of Web services is more than just traffic parameters. Beyond the network aspects, QoS for Web services covers server performance, security, transactional, and monetary aspects, and all components and layers participating in the Web service communication process.

This thesis targets mainly the following issues in Web service communication:

1. The definition of QoS aspects and parameters related to the Web service layer.
2. The efficient lookup and selection of services at runtime according to clients' requirements.
3. The mapping of QoS aspects and parameters, which are defined in the Web service layer, on the underlying communication layer and on the participating components.
4. Efficient access of mobile clients to Web services

1.2 Contributions

The main contributions of this thesis are the design and performance measurements of the WS-QoS framework targeting the overall QoS support for

Web services. Our WS-QoS framework is based on the WS-QoS XML schema that allows service clients and service providers to define QoS-aware requirements and offers. Furthermore, the WS-QoS XML schema allows domains and components along the Web service communication process to actively support the clients' QoS requirements. The flexible and extensible WS-QoS framework addresses various aforementioned QoS aspects, not only the classical network aspect.

We have implemented our WS-QoS framework. We have conducted performance measurements of our framework. The measurement results prove the advantages of applying our framework for QoS aware Web service communication.

Another outstanding part of this work in comparison to other Web service related efforts is that we consider QoS through different layers and components that participate in Web service communication. Users can define their QoS requirements due to various aspects on a higher level such as in the application layer by applying the WS-QoS XML schema. These QoS aspects and their QoS parameters are evaluated and mapped at runtime to achieve QoS fulfillments.

Moreover, this thesis presents solutions to support mobile Web services by applying the WS-QoS framework as well as solutions to protect web servers hosting Web services from overloading. The prototype implementations and the performance measurements of the framework and solutions prove the feasibility and advantages of the WS-QoS framework.

1.3 Thesis overview

This thesis is structured as follows:

Chapter 2 introduces background information on Web services, QoS issues, and mobile Web services. Its first part introduces the Web service protocol stack and Web service related protocols. The second part discusses QoS metrics and aspects. The third part introduces the notion of mobile Web services and constraints of mobile devices.

Chapter 3 gives an overview of selected major industrial and academic approaches towards QoS specification and management for Web services. After discussing the state of the art of the Web service technology, five examples will give an impression of the variety of research in this area.

Chapter 4 describes the design of the WS-QoS framework, which is fully compatible to standard Web services protocols such as SOAP, WSDL, and UDDI. The specific elements of the WS-QoS framework are integrated into WSDL in a standard conform way. The fundamental goal of the design of the WS-QoS architecture is QoS support during the whole communication process. Our framework supports standard conformity, scalability, extensibility as well as QoS mapping between different layers in terms of the Internet model.

Section 4.1 examines the requirements for QoS-aware Web service communication. Section 4.2 presents the WS-QoS XML schema that is applied to

define both QoS requirements and offers. Section 4.3 focuses on the QoS-aware service discovery and selection based on the QoS requirements and offers. Section 4.4 discusses the design issues for QoS-aware service invocation. A discussion and evaluation of the architecture presented in the chapter can be found in 4.5.

Chapter 5 presents the prototypic implementation of the WS-QoS framework introduced in Chapter 4. The implementation supports both wired and mobile devices. It encompasses the Web Service Broker (WSB), the Requirement Manager, and the Base and Supporting Functions.

Chapter 6 explains how to apply the implementation of the WS-QoS framework. This chapter describes three implementation issues: First, the service should implement a generic service interface. Second, the service has to implement a strategy to provide WS-QoS offers, which should be adjustable to changing situations of service utilization. Finally, to achieve the QoS level(s) associated with distinct offers, the selected offer and further QoS requirements have to be evaluated when receiving a request.

Chapter 7 presents the performance measurements of our framework and implementation. In the first measurement, we determine the impact of Web service overhead on web servers and mobile clients. We present our solutions that improve the performance of mobile Web service access. In the second measurement, we demonstrate the advantages of the WSB when the lookup and selection of a Web service offer from many competing offers is processed at runtime. In the third measurement, we demonstrate the performance gain achieved through our QoSProxy that maps clients' QoS requirements of transport network at runtime. In the last measurement, we demonstrate the advantage of our WS-QoS framework that prevents Web service servers from becoming overloaded with adaptive WS-QoS offers.

Chapter 8 summarizes this thesis and discusses options for future research.