Mobile Web Service Provisioning – QoS and Discovery Issues

Satish Narayana Srirama[1], Matthias Jarke[1,2], and Wolfgang Prinz[1,2]

[1] Information Systems Group, RWTH Aachen University
Ahornstr. 55, 52056 Aachen, Germany
[2] Fraunhofer FIT
Schloss Birlinghoven, 53754 Sankt Augustin, Germany
{srirama, jarke}@cs.rwth-aachen.de
wolfgang.prinz@fit.fraunhofer.de

Abstract. The advanced features of today's smart phones and hand held devices, like the increased memory and processing capabilities, allowed them to act even as information providers. Thus a smart phone hosting web services is not a fancy anymore. During one of our projects, we have developed such a Mobile Host and analysed its performance and application scope. While Mobile Host is technically feasible, the ability to provide proper QoS in the vulnerable and volatile mobile ad-hoc topologies is quite challenging. The QoS should ensure secure and reliable web service communication in the resource constrained radio link. While many standards exist in the wired network, ensuring the QoS of web services, not much has been analysed and standardized for mobile web services. Our study contributes to this work and we tried to adapt some of the existing QoS standards to mobile web services domain.

Mobile Host opens up a new set of applications and it finds its use in many domains like mobile community support, collaborative learning, social systems etc. But the relevant discovery of the services provided by the smart phones has became quite complex, because of the volume of services possible with each Mobile Host providing some services. Centralized registries have severe drawbacks in such a scenario and alternate means of service discovery are to be addressed. P2P domain with its resource sharing capabilities comes quite handy and here in this paper we provide an alternate approach to centralized registry for discovering mobile web services. The services are published into the P2P network as JXTA modules and the discovery issues of these module advertisements are addressed. The P2P discovery approach also provides alternate means of identifying and addressing the Mobile Host.

Keywords: Mobile web service provisioning, QoS, WS-Security, peer to peer (P2P), JXTA and mobile web service discovery.

1 Introduction

It is well accepted by now that the Internet can be seen as a large-scale distributed information system with numerous information providers and users. From the

information systems engineering's view-point, the Internet has lead the evolution from static content to web services. Web services are distributed software components which can be accessed over the Internet using well established web mechanisms and XML-based open standards and transport protocols such as SOAP [1] and HTTP [2]. Public interfaces of web services are defined and described using W3C based standard, Web Service Description Language (WSDL) [3], regardless of their platforms and implementation details. Web services have wide range of applications and primarily used for enterprise integration. The biggest advantage of web services lies in their simplicity in expression, communication and servicing. The componentized architecture of web services also makes them reusable, thus reducing the development time and costs. [4]

Concurrently, the capabilities of high-end mobile phones and PDAs have increased significantly, both in terms of processing powers and memory capabilities. The smart phones are becoming pervasive and are being used in wide range of applications like location based services, mobile banking services, ubiquitous computing etc. The market capture of such smart phones is quite evident and in fact in 2003 itself 12.1 million PDA-sized devices were sold, including all PDA-phones and smart phones. The number of Java enabled mobile phones sold, in the same time, has outnumbered the number of PCs sold [5]. The higher data transmission rates achieved in wireless domains with 3G [6] and 4G [7] technologies and the fast creeping of all-ip broadband based mobile networks also boosted this growth in the cellular market. The situation brings out a large scope and demand for software applications for such high-end smart phones. The statistic analysis provided by Idealliance [5] further enhances the point that in the very near future; best part of the requirement will end up at a device called high-end mobile phone/handset.

To meet this demand of the cellular domain and to reap the benefits of the fast developing web services domain and standards, the scope of the mobile terminals as both web services clients and providers is being observed. Mobile web services enable communication via open XML web service interfaces and standardized protocols also on the radio link, where today still proprietary and application- and terminal-specific interfaces are required. To support the mobile web services, there exist many organisations such as OMA [8], LA [9] on the specifications front; some practical data service applications such as OTA (over-the-air provisioning), application handover etc. on the commercial front; and SUN, IBM toolkits [10], [11] on the development front. Thus, though this is early stages, we can safely assume that mobile web services are the road ahead. Mobile web services lead to manifold opportunities to mobile operators, wireless equipment vendors, third-party application developers, and end users. While mobile web service clients are common these days [12], we have studied the scope of mobile web service provisioning, in one of our previous projects. In this project, we have developed a Mobile Host [13], capable of providing basic web services from smart phones. Once the Mobile Host was developed, extensive performance analysis was conducted to prove its technical feasibility. [14]

While service delivery and management from Mobile Host are technically feasible, the ability to provide proper Quality of Service (QoS), especially in terms of security and scalability, for the Mobile Host is observed to be very critical. In terms of security, the Mobile Host has to provide secure and reliable communication in the

vulnerable and volatile mobile ad-hoc topologies. Moreover with the easily readable mobile web services, the complexity to realize security increases further. For the traditional wired networks and web services, a lot of standardized security specifications, protocols and implementations like WS-Security [15], SAML [16] etc., exist, but not much has been explored and standardized in wireless environments. Some of the reasons for this poor state might be the lack of widely active commercial data applications, to-date. Our study contributes to this work and tries to bridge this gap, with main focus at realizing some of the existing security standards in the mobile web services domain. In this study we have analyzed the adaptability of WS-Security to the mobile web service provisioning domain. Mainly we observed the latency caused to performance of the Mobile Host, with introduction of security headers into the exchanged SOAP messages. The performance penalties of different encryption and signing algorithms were calculated, and the best possible scenario for securing mobile web services communication is suggested. [17]

In terms of scalability, the layered model of web service communication, introduces lot of message overhead to the exchanged verbose XML based SOAP messages. This consumes lot of resources, since all of this extra information is to be exchanged over the radio link. Many compression techniques are being studied to reduce the size of messages being exchanged in mobile web service communication. But this approach comes with a trade-off that, now the compressed techniques need some extra processing power at the smart phones and thus adding further performance latencies.

Mobile Host opens up a new set of applications and it finds its use in many domains like mobile community support, collaborative learning, social systems etc. Primarily, the smart phone can act as a multi-user device without additional manual effort on part of the mobile carrier. Many applications were developed and demonstrated, for example in a distress call; the mobile terminal could provide a geographical description of its location (as pictures) along with location details. The Mobile Host in a cellular domain is of significant use in any scenario which requires polling that exchanges significant amount of data with a standard server, for example a mobile checking for the updates of RSS feeds provided by a server.

While the applications possible with mobile web services are quite welcoming, the huge number of web services possible, with each Mobile Host providing some services in the wireless network, makes the discovery of these services quite complex. Proper discovery mechanisms are required for successful adoption of mobile web services into commercial environments. The traditional centralized UDDI based registries [18] have many limitations in this aspect and might not be the perfect solution for the mobile web service discovery. The dynamic nature of the mobile nodes further enhances this problem. Considering these difficulties, we are proposing an alternative approach of discovering mobile web services. The method uses the peer to peer (P2P) [19] network for advertising the web services and depends on the network for discovering the services. We have developed the solution using the JXTA network [20] and its features, and were able to publish mobile web services and discover them from the smart phones, with reasonable performance latencies.

In summary, the paper discusses our project “Mobile Web Service Provisioning“, with its QoS issues, challenges and difficulties in inducing the current existing QoS standards into mobile web services domain. The paper also discusses the issues with

mobile web service discovery and tries to provide alternate solution for centralized registry based web service discovery mechanism. The rest of the paper is organized as follows:

Section 2 discusses the concept, performance analysis and applications of mobile web service provisioning. Section 3 addresses the QoS issues considering both the security and scalability challenges in the mobile web services domain. The section describes existing and emerging standards in mobile and web services domains and discusses some of the QoS realization details and their analysis on our Mobile Host. Section 4 discusses the challenges with mobile web service discovery and tries to provide alternatives for centralized registries, with the adaptation of Mobile Host into the JXTA network. Section 5 concludes the paper with future research directions.

2 Mobile Web Service Provisioning

Service Oriented Architecture (SOA) [21] is the latest trend in information systems engineering. It is a component model, presenting an approach for building distributed systems. SOA delivers application functionality as services to the end-user applications and other services, bringing the benefits of loose coupling and encapsulation to the enterprise application integration. SOA defines participating roles as, service provider, service client, and service registry. SOA is not a new notion and many technologies like CORBA [22] and DCOM [23] at least partly represent this idea. Web services are newest of these developments and by far the best means of achieving SOA. Using web services for SOA provides certain advantages over other technologies. Web services are based on a set of still evolving, though well-defined W3C standards, that allow much more than, just defining interfaces.

The web service architecture [4] defined by the W3C enables application-to-application communication over the Internet. Web services are self-contained, modular applications whose public interfaces are described using Web Services Description Language (WSDL). Web services allow access to software components through standard Web technologies and protocols like SOAP and HTTP, regardless of their platforms, implementation details. A service provider develops and deploys the service and publishes its description and binding/access details (WSDL) with the UDDI registry [18]. Any potential client queries the UDDI, gets the service description and accesses the service from service provider using SOAP. The communication between client and UDDI registry is also based on SOAP. [24]

Web services and its protocol stack are based on open standards and are widely accepted over the internet community. Web services have wide range of applications and range from simple stock quotes to pervasive applications using context-awareness like weather forecasts, map services etc. The biggest advantage of web services lies in its simplicity in expression, communication and servicing. The componentized architecture of web services also makes them reusable, thereby reducing the development time and costs.

The quest for enabling these open XML web service interfaces and standardized protocols also on the radio link, with the current developments in cellular domain, lead to new domain of applications mobile web services. Traditionally, the hand-held

cellular devices have many resource limitations like limited storage capacities, low computational capacities, and small display screens with poor rendering potential. Most recently, the capabilities of these wireless devices like smart phones, PDAs are expanding quite fast. This is resulting in quick adoption of these devices in domains like mobile banking, location based services, social networks, e-learning etc. The situation also brings out a large scope and demand for software applications for such high-end wireless devices. Moreover, higher data transmission rates, in the order of few Mbs, were achieved in cellular domain, with interim and third generation mobile communication technologies like GPRS [25], [26], EDGE [27] and UMTS [28]. Most recently with the advent of 4G technologies and their deployment in south Asian countries suggests that mobile data transmissions of the rate of few Gbs is also possible [29].

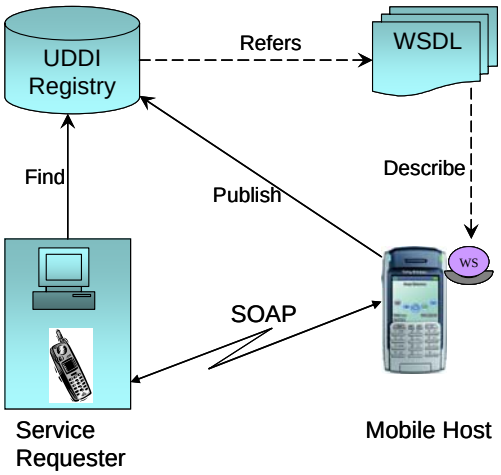

Fig. 1. Basic mobile web services framework with the Mobile Host.

In mobile web services domain, the resource constrained mobile devices are used as both web service clients and providers. Web services have a broad range of service distributions and on the other hand cellular phones have large and swiftly expanding user base. Combining these two domains brings us a new trend and lead to manifold opportunities to mobile operators, wireless equipment vendors, third-party application developers, and end users. While mobile web service clients are common these days, and many software tools [10], [11] are already existent in the market, easing their development and adoption, the research with providing web services from smart phones is still sparse. In our mobile web service provisioning project one such Mobile Host was developed and its performance was extensively analyzed, proving the feasibility of concept. Figure 1 shows the basic mobile web services framework with web services being provided from the Mobile Host.

2.1 Architecture and Implementation Details of Mobile Host

Mobile Host is a light weight web service provider built for resource constrained devices like cellular phones. Though it was developed for resource constrained devices, it followed the same architecture as general web services. The Mobile Host has been developed as a web service handler built on top of a normal web server. Mobile web service messages can be exchanged using the SOAP [1] over different transportation protocols like HTTP, UDP, and WAP etc. In our Mobile Host's implementation the web service requests sent by HTTP tunneling are diverted and handled by the web service handler. The key challenges addressed in Mobile Host's development are threefold: to keep the Mobile Host fully compatible with the usual web service interfaces such that clients will not notice the difference; to design the Mobile Host with a very small footprint that is acceptable in the smart phone world; and to limit the performance overhead of the web service functionality such that neither the services themselves nor the normal functioning of the smart phone for the user is seriously impeded.

Figure 2 shows the core architecture of the Mobile Host. At the HTTP interface, the Mobile Host listens for incoming HTTP GET/POST requests on a sever socket. When the Mobile Host receives a request, the server socket accepts it, creates a socket for communication, and initiates a new thread of execution by creating an instance of the request handler. The request handler extracts the incoming message from the input stream of the socket, and checks for web service requests sent via HTTP tunneling. If it is normal HTTP request, the request handler processes the HTTP request just as the standard web server, and returns the response by writing the message to the output stream of the socket.

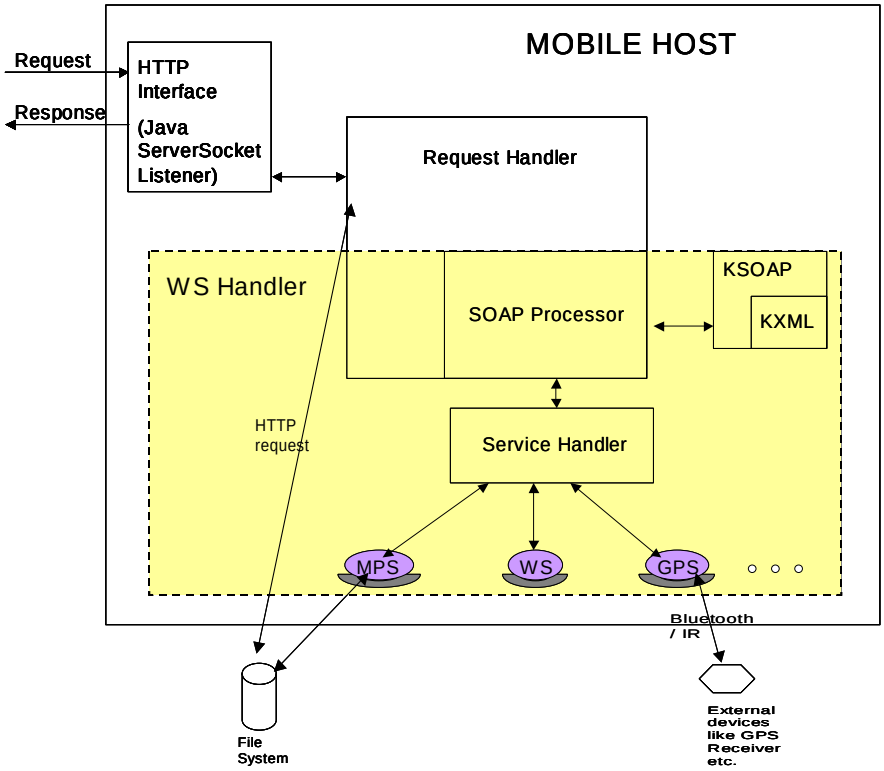

Fig. 2. Core architecture of the Mobile Host

If the message comprises a web service request, the Web Service Handler component of the Mobile Host processes the message. The request handler reads the HTTP message body and de-serializes the SOAP request to Java objects, using the SOAP processor. The request handler passes these objects to the service handler, which extracts the service details and invokes the respective service. The business logic of the service method is then executed and the service handler returns the response to the request handler. The web services deployed on the Mobile Host can access the local file system, or any external devices like a GPS receiver, using Infrared, Bluetooth etc., and can implement business logic. The request handler serializes the response and prepares the HTTP response message, which is returned to the client as a HTTP response by writing to the output stream of the socket.

The Mobile Host was developed in PersonalJava [30] on a SonyEricsson P800 smart phone. The footprint of our fully functional prototype is only 130 KB. Open source kSOAP2 [31] was used for creating and handling the SOAP messages. kSOAP2 is thin enough to be used for resource-constrained devices and provides a SOAP parser with special type mapping and marshalling mechanisms. Considering the low-resource constraints of smart phones, no deployment environment can be easily provided. Hence, all services have to be deployed at the installation of the Mobile Host. Alternatively, the Mobile Host was configured to look for services at

other locations apart from the main JAR location, where the services could then be deployed at runtime. Similar implementations of Mobile Host are also possible with other Java variants like J2ME [32], for smart phones. We also have developed a J2ME based Mobile Host and its performance was observed to be not so significantly different from that of the PersonalJava version.

Even though the web service provider is implemented on the smart phone, the standard WSDL can be used to describe the services, and the standard UDDI registry can be used for publishing and un-publishing the services. An alternative for the UDDI-based discovery is being studied, where we are trying to realize Mobile Host in a Peer to Peer (P2P) [33] network, there by leveraging the advertising and searching of WSDL documents to the P2P network [34]. The approach is addressed in later sections of this paper, while discussing the discovery issues of mobile web services.

Alternate architectures for mobile web service provisioning are also possible with SOAP compliant proxy or gateway in between the Mobile Host and the web service requester. The communication between the client and the proxy is using SOAP and the communication between the proxy and the Mobile Host would be using a protocol, efficient for the data transport across the mobile networks. Many such proprietary protocols and implementations have evolved like WSOAP [35], gSOAP [36], eSOAP [37], etc. Wireless SOAP (WSOAP) [35] is basically a set of optimization techniques. The WSOAP aims to provide static encoding based on SOAP schema, leverages WSDL service description to create adaptive encoding for web service interfaces, limits computational cost, and concentrates on functional message equivalence rather than exactness. This protocol can be extremely useful between mobile devices and gateways where the resources are very limited as WSOAP can reduce SOAP message sizes by 3-12 times.

The gSOAP toolkit is a platform-independent development environment for C and C++ web services [36]. gSOAP provides transparent SOAP API through the use of compiler technology that hides irrelevant SOAP-specific details from the user. The compiler automatically maps native and user-defined C and C++ data types to semantically equivalent SOAP data types and vice-versa. As a result, full SOAP interoperability is achieved with a simple API. Similarly, eSOAP is small lightweight implementation of the SOAP specifically designed for embedded systems. The eSOAP toolkit is a C++ library that provides a SOAP processing engine for the embedded system [37].

We are considering this option of using SOAP proxy in our study, which handles the security and the scalability issues, there by providing better QoS for the Mobile Host. For scalability issues of Mobile Host, compression technologies with Fast Web Services [38] are being considered, instead of proprietary SOAP implementations. SOAP compression is highly efficient in the mobile web services domain, because of the poor connectivity and high communication costs of the mobile networks.

2.2 Mobile Terminal Access

Once a mobile web service is developed and is deployed with the Mobile Host, the mobile terminal, that is registered and connected within the mobile operator network, requires some means of identification and addressing, which allows the web service to

be accessible also from Internet. Generally, computers and devices in a TCP/IP network are identified using an IP address. The IP address, that is required for the data transfer to and from smart phones (as for any other IP communication client as Web servers, Intranet workstations, etc.), is assigned during the communication configuration phase. Typically, the IP address assigned to mobile devices using GPRS is only temporarily available, and is known only within the mobile operator's network, which makes it difficult to use the IP address in the client applications.

Our study has identified different means of resolving the IP address in HSCSD [39] (High-Speed Circuit Switched Data) dial-up connection, GPRS [26] (General Packet Radio Service) environments and thereby making the data transmission with a mobile terminal, possible. Here we discuss two of these identified methods.

HSCSD.

Figure 3 illustrates the architecture used to connect the Mobile Host to the prototyping network using a HSCSD dial-up connection. In this architecture a HSCSD connection is established between the mobile terminal and the prototyping network, which is connected to the Internet. The connection uses a Public Land Mobile Network (PLMN) and the Public Switch Telephone Network (PSTN / ISDN) for making the data call to the server. The connection is setup by using PPP (Point-to-Point Protocol) over a circuit-switched data call to a modem that is connected to one of the servers in the network. On top of this PPP link a TCP/IP end-to-end connection between the mobile terminal and the dial-in server is established. Hence, as long as the data call persists, the mobile terminal can be addressed using the IP address assigned to it by the dial-in server. Thus the web service deployed on the mobile terminal can be accessed from any client within the network environment.

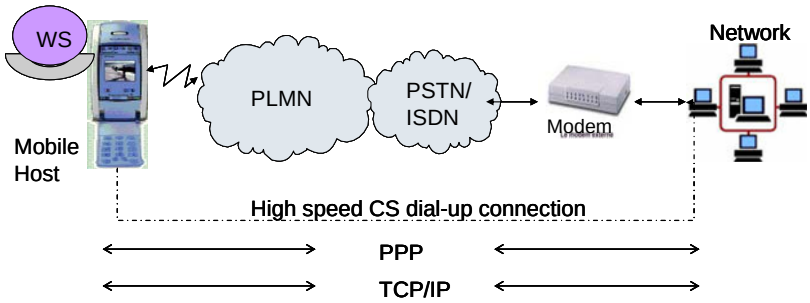

Fig. 3. Architecture for an end-to-end TCP/IP connection between the mobile terminal and the prototyping network using HSCSD connection.

Using an appropriate NAT configuration, the mobile web service can be accessed by any service requestor from the Internet. Using the NAT, the network provides a DNS name for the Mobile Host. The only requirement towards the PPP daemon is that the mobile terminal should always receive the same IP address when it connects to the dial-in server.

The main drawback with the HSCSD solution is the circuit switched connection, which would have to persist as long as the Mobile Host should be available for the

access of its web services. The billing of circuit switched data connection is based on the time the connection persists, not on the amount of data transmitted across the network. This makes this scenario unfeasible for commercial purposes. Volume based charging is a major advantage enabled by GPRS.

GPRS.
Once the GPRS connection is established the mobile can be identified by the temporary IP provided by the mobile operator network. It is also feasible to have a public IP for the mobile terminal, a feature provided by very few operators today. The operational setup for accessing the mobile terminal in a GPRS network is given in figure 4. The mobile TCP/IP connection between the web service client and the Mobile Host is deployed on top of a GPRS link into the mobile operator network. From there the traffic is routed through the Internet to/from the web service client.

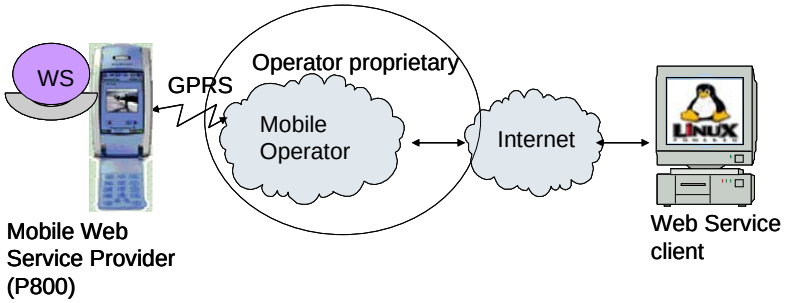

Fig. 4. The operational setup of Mobile Host in a live GPRS environment.

The problem of addressing each mobile node with IP is not a big issue and it could be solved with Mobile IP version 6 (Mobile IPv6) [40]. But the necessity of public IP for each smart phone is observed to be major hindrance for the commercial success of Mobile Host, even though IPV6 promises such availability. Alternatives for the mobile terminal access are studied and we will address one such solution using the P2P network, when we will be discussing the discovery issues of mobile web services.

2.3 Sample Web Services Provided by Mobile Host

Before considering the performance analysis of the Mobile Host and its applications, we describe some of the basic web services provided from our Mobile Host. These services give an idea of some of the services possible from smart phones and were used in calculating the performance loads of the Mobile Hosts.

Mobile Photo Album Service.
Today's high-end mobile terminals become more and more advanced, and are generally being equipped with an integrated digital camera. The photographs taken with these smart phones can later be uploaded or transferred to PCs through cables or

by using wireless methods like Infrared or Bluetooth. Using currently available technologies, if a user wants to publish the photographs he had taken with the mobile terminal to the public or friends, he has to upload the photos to a Web server, from which they can be accessed. The user can also send the images through Multimedia Messaging Service (MMS) [41] or some other means of messaging to the clients. Here the mobile owner bears the payment for the communication between his smart phone and the Web server or the receiver's device. With a mobile web service provider, implemented and deployed on the smart phone, interested people can access the Mobile Host using a standard web service client or a Web client, and can browse through the pictures they are interested in. Here the responsibility for payment shifts to the actual clients, who are browsing the pictures provided by the Mobile Host. The service is comparable to any other online image album service or blog service, but implemented on the mobile terminal.

Location (GPS) Data Provisioning Service
This dedicated web service provides the exact location information of the mobile terminal, such as GPS (Global Positioning System) data [42]. The service uses a Socket GPS receiver for getting the GPS co-ordinates. The external device is connected to the smart phone via Bluetooth. The GPS data can also be collected while taking the pictures and these two details can be mapped together, giving scope for many interesting scenarios like the traveller's diary etc. The GPS co-ordinates can always be mapped to geo spatial maps.

2.4 Performance Analysis of the Mobile Host

Once the Mobile Host was developed, it was extensively tested for performance issues like the memory load, server-processing load etc. The evaluation of the system was conducted using services already described like the mobile photo album service, the location (GPS) data provisioning service and some more basic services like echo, 'ls' services and etc.

The test setup comprised a Mobile Host developed and deployed on the P800 smart phone and a standalone Apache Axis [43] web service client. The client invokes different services (Within the context of this discussion, it is assumed that the client knows the exact location (URI) of the service and the service description;) deployed on the Mobile Host and the performance of the Mobile Host was observed, by taking timestamps and memory foot prints, while the Mobile Host was processing the web service request. The tests were conducted both in HSCSD and GPRS environments.

As the test cases for the mobile photo album service, 15 different images were selected with memory sizes ranging from 3Kb to 100Kb. The client tried to browse through these pictures. The location (GPS) data provisioning service uses an external GPS device for providing the GPS data.

The detailed performance evaluation of the Mobile Host clearly showed that service delivery as well as service administration can be performed with reasonable ergonomic quality by normal mobile phone users. Figure 5 shows the time delays of different activities, for the location data provisioning service. As the most important result, it turns out that the total web service processing time at the Mobile Host is only

a small fraction of the total request-response invocation cycle time (<10%) and rest all being transmission delay, in a GPRS network. This makes the performance of the Mobile Host directly proportional to achievable higher data transmission rates. [13]

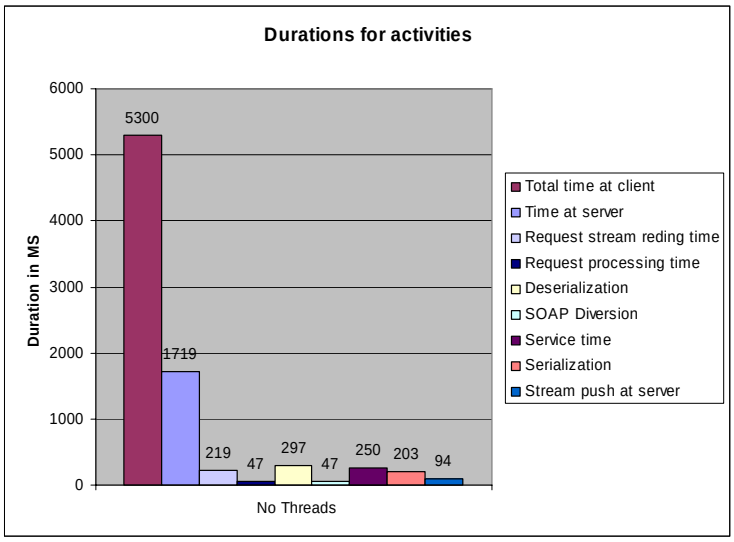

Fig. 5. Time stamps for the GPS data provisioning service.

The second-generation GSM networks delivered high quality and secure mobile voice and data services like SMS (Short Message Service) [44], circuit switched Internet access etc., with full roaming capabilities and across the world. The GSM platform is a widely successful wireless technology and it was the world's leading mobile standard. But, with the advent of the interim-generation technologies like GPRS and EDGE, and third-generation technologies like UMTS, still higher data transmission rates are achieved in the wireless domain, in the order of few hundreds of Kbs to 2 Mbs. Most recently with the advent of 4G technologies and their deployment in south Asian countries suggests that mobile data transmissions of the rate of few Gbs is also possible [29]. These developments make the Mobile Host soon realizable in commercial environments and applications.

In terms of performance of the Mobile Host, the key question was whether a reasonable number of clients could be supported with an overhead that would not prevent the main mobile user from using his or her smart phone in the normal fashion (either to supply the services or just for usual local phone functions). This study was also required since it would define the limit for the number of concurrent participants in the collaborative application environments. The Mobile Host was successful in handling up to 8 concurrent accesses for reasonable services like location data provisioning service with response size of approximately 2Kb.

2.5 Applications of the Mobile Host

Mobile Host opens up a new set of applications and it finds its use in many domains like mobile community support, collaborative learning, social systems etc. Primarily, the smart phone can act as a multi-user device without additional manual effort on part of the mobile carrier. Many applications were developed and demonstrated, for example in a distress call; the mobile terminal could provide a geographical description of its location (as pictures) along with location details.

Another interesting application scenario involves the smooth co-ordination between journalists and their respective organizations. The scenario is illustrated in figure 6. Journalists can be at different locations across the globe, covering different events like the sport events, conferences etc. An editor can always keep track of the location of "his" journalists and the content they have gathered. He can browse through the pictures taken by the journalist at any instance. Standard client applications can be developed for the editor, which synchronize the information stored by editor and data at the Mobile Host. The key difference to the more traditional solutions where journalists upload their contents to a server held by the editor is that parallel access to the Mobile Host by both the journalist and the editor is possible; even other journalists in the team can look at the mobile information thus better synchronizing their activities, e.g. in the coverage of some major distributed event. Thus, the journalists can concentrate more on their job of collecting, as they don't have to upload the data, every time they get something interesting. The data can later be synchronized with a server for archives, when the journalists are free and off the event site. [14]

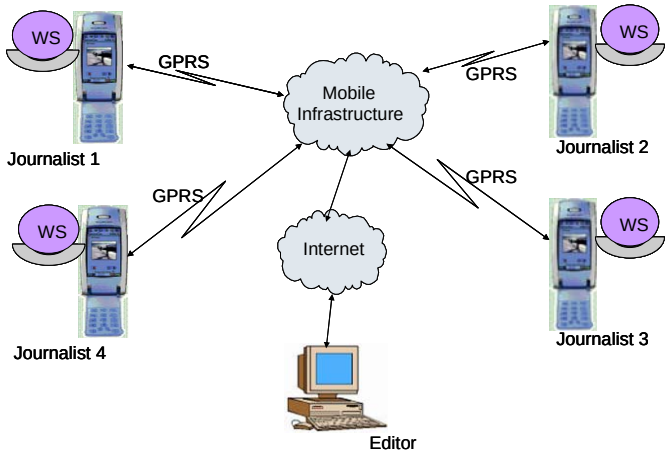

Fig. 6. The Mobile Host in collaborative journalism scenario.

Most recently the scope of the Mobile Host in m-learning (mobile learning) domain is also being studied. As the Mobile Host, the mobile terminal can provide access to information like pictures, audios, videos, tags, documents, location details, and other learning services [45]. Many m-learning application scenarios can be envisioned, like podcasting, mobile blogging, mobile learning media sharing service,

expertise finder service etc. In the mobile learning media sharing scenario, learners can share audio or video lecture recordings or go for the field study and take the pictures of the location. Peers can then browse through the pictures taken, add tags, and give their suggestions or comments. In an expertise finder learners can look for reliable access to learning resources, persons who share the same interests, and experts with the required know-how that can help achieving better results. In the e-learning aspect these experts can share the information among the other users. Examples of these use cases could be exchanging the mathematical formulas [46] and the experts validating them or even correcting them. The Mobile Host in a cellular domain is of significant use in any scenario which requires polling that exchanges significant amount of data with a standard server, for example a mobile checking for the updates of RSS feeds provided by a server. The Mobile Host can eliminate polling process as the RSS feeds can now be directly sent to the Mobile Host, when the RSS feeds are updated.

From the commercial viewpoint, with the Mobile Host, there can be a reversal of payment structures in the cellular world. While traditionally the information-providing web service client has to pay to upload his or her work results to a stationary server (where then other clients have to pay again to access the information), in the Mobile Host scheme responsibility for payment can be shifted to the actual clients -- the users of the information/services provided by the Mobile Host. Thus Mobile Host renders possibility for small mobile operators to set up their own mobile web service businesses without resorting to stationary office structures [13].

The Mobile Hosts in an operator proprietary network can also form a P2P network with other mobile phones and can share their individual resources and services. P2P offers a large scope for many applications with Mobile Host. Not just the enhanced application scope, the P2P network also offers better identification and discovery mechanisms of huge number of web services possible with Mobile Hosts [34]. We will discuss the approach in mobile web service discovery section.

While Mobile Host is technically feasible and its applications are quite welcoming in different domains, the ability to provide proper QoS in the vulnerable and volatile mobile ad-hoc topologies is quite challenging. The QoS should ensure secure and reliable communication. Secure provisioning of mobile web services needs proper message-level security consisting data integrity, confidentiality and end-point access security that constitutes authentication, authorization and access control. In terms of scalability of the Mobile Host, the key question is whether a reasonable number of services could be provided by the smart phone in the cellular domain.

3 QoS Aspects in Mobile Web Services Domain

While service delivery and management from Mobile Host are technically feasible, and its applications are quite welcoming in different domains, the ability to provide proper Quality of Service (QoS), especially in terms of security and scalability, for the Mobile Host is observed to be very critical. In terms of security, the Mobile Host has

to provide secure and reliable communication in the vulnerable and volatile mobile ad-hoc topologies. Moreover with the easily readable mobile web services, the complexity to realize security increases further. For the traditional wired networks and web services, a lot of standardized security specifications, protocols and implementations like WS-Security, SAML etc., exist, but not much has been explored and standardized in wireless environments. Some of the reasons for this poor state might be the lack of widely active commercial data applications, to-date. Our study contributes to this work and tries to bridge this gap, with main focus at realizing some of the existing security standards in the mobile web services domain. In terms of scalability, the layered model of web service communication, introduces lot of message overhead to the exchanged verbose XML based SOAP messages. This consumes lot of resources, since all of this extra information is to be exchanged over the radio link. Many compression techniques are being studied to reduce the size of messages being exchanged. But this approach comes with a trade-off that, now the compression techniques need some extra processing power at the smart phones and thus adds further performance latencies.

In this section we will discuss our security and scalability analysis of mobile web service provisioning. We have analyzed the adaptability of WS-Security to the mobile web service provisioning domain. Mainly we observed the latency caused to performance of the Mobile Host, with the introduction of security headers into the exchanged SOAP messages. The performance penalties of different encryption and signing algorithms were calculated, and the best possible scenario for securing mobile web services communication is suggested. We will also discuss briefly about our ongoing scalability research.

3.1 Security Challenges for Mobile Web Services

Once the web services are deployed with the Mobile Host, the services are prone to different types of security breaches like denial-of-service attacks, man-in-the-middle attacks, intrusion and spoofing etc. Mobile web services use message-based technologies (SOAP over HTTP) for complex transactions across multiple domains. SOAP by itself does not specify the means of providing the security for the web service communication. Also many legitimate intermediaries might exist in the web service communication making the security context requirement to be from end-to-end. Hence the traditional point-to-point security technologies like the SSL [47], HTTPS [48] and full encryption provided by the 3G technologies like UMTS communication technology can’t be adapted for the mobile web services domain. These methods also affect the transportation independency feature of the SOAP messages by restricting the messages to particular transportation protocols. Hence, the need for sophisticated end-to-end message-level security becomes a high priority for mobile web services.

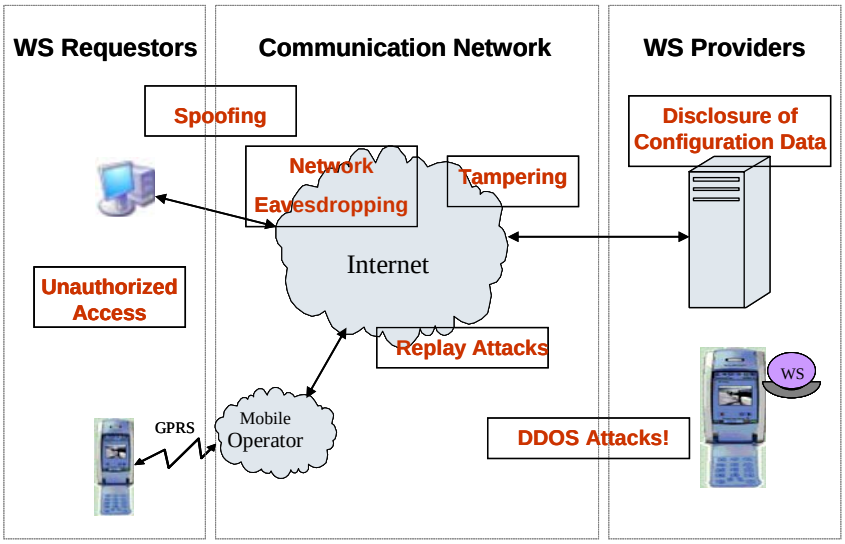

Fig. 7. Typical security breaches in the mobile web services.

Figure 7 depicts some of the typical security breaches in web service and wireless environments, across the mobile web services domain [49]. Spoofing is a means of accessing a system with false identity. To accomplish this, an attacker can use stolen user credentials or fake source address that does not represent the actual source address. The purpose of spoofing would be to hide the original source of an attack or to gain access to a service as a legitimate user or host, thereby acquiring sensitive privileges. Proper authentication and authorization principles are to be used to cover spoofing and unauthorized access.

Tampering is an act of unauthorized modification of the web service message in the network by any intermediary. Mobile web services are very prone to this attack as there might be many legitimate intermediaries in the web service communication and an attacker can spoof any of the intermediaries. Network eavesdropping or sniffing is the process of monitoring traffic for sensitive data such as plaintext passwords or configuration information by placing packet sniffers in the middle of the network. Proper encryption and digital signatures help in avoiding tampering and network eavesdropping attacks.

Replaying a valid, changed or unchanged message to a web service by impersonating the client is referred as replay attack. The unchanged message replay attack also known as basic replay attack can be avoided by using nonce, a cryptographically unique value, with the web service message. But the most common types of message replay attacks are man in the middle attacks where the attacker captures the message, changes the contents and replays them to the web service. Proper encryption and digital signatures again help in preventing this form of replay attack.

Denial-of-service (DOS) is a process of making a system, server or application unavailable, by overloading the system. For each individual service, maintaining and understanding the collection of data can help in protecting it from denial-of-service

attacks. But having such a scenario implemented on the resource constrained mobile phones could be impractical. Security policies and high-level access control mechanisms should help to a certain extent in this regard.

Last but not the least of the security breaches shown in figure 7 is the disclosure of configuration data. Generally, WSDL documents reveal lot of information about web services and other sensitive information like configuration data of servers. Proper and authorized access of WSDL documents is to be allowed, to avoid these unwanted disclosures.

Considering the security breaches in the mobile web services, the mobile web service communication should support at least the basic security requirements as emphasized in figure 8. Secured message transmission is achieved by ensuring confidentiality and data integrity, while authentication and authorization will ensure that the service is accessed only by the trusted service requestors. Upon successful application of these basic security requirements, trust and policy can be considered for mobile web services domain. Policy and trust ensure proper choreography of services. Policy defines general security policy assertions over web service security whereas Trust builds trust relationships on web services security for exchanging security tokens by providing a proper framework.

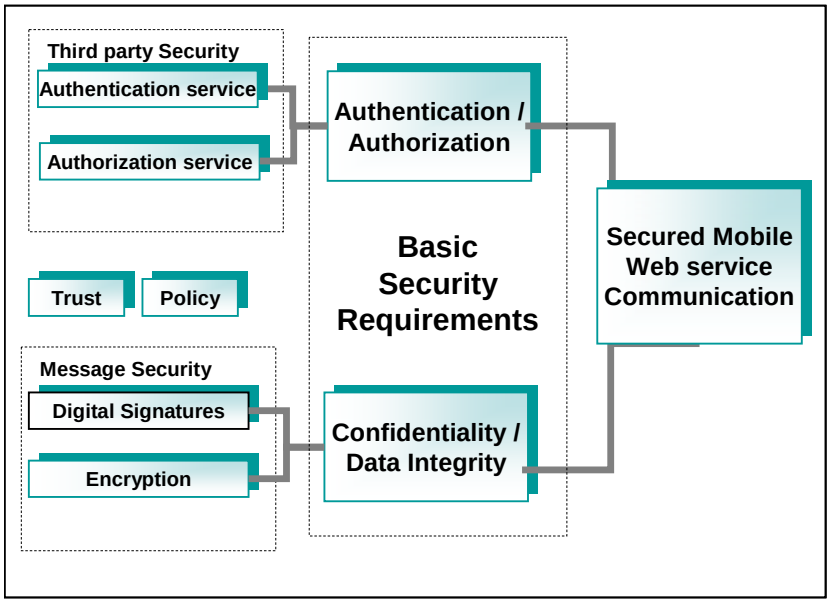

Fig. 8. Basic security requirements for mobile web services.

3.2 Existing and Emerging Mobile Web Service Security Standards

Before considering the analysis and circumvention of security challenges for the mobile web services domain, this section discusses briefly the existing security standards and specifications and some relevant notable projects in web services and wireless domains. Listed below are some of the standard committees and organizations working around web services, wireless domain and their security thus helping in mobile web services domain security:

- W3C is primarily responsible for SOAP, XML Encryption [50], XML Signature [51] and WSDL standards.
- OASIS is an organization which has larger interest in web service specific standards and it owns primary areas of our interest such as WS-Security [15] and SAML [16] standards.
- Liberty Alliance (LA) [9] group was aimed at providing a framework for interoperable federated identity.
- Open Mobile Alliance (OMA) [8] was formed to develop and promote interoperability for mobile data services.

WS-Security.
The WS-Security specification from OASIS is the core element in web service security realm. It provides ways to add security headers to SOAP envelopes, attach security tokens and credentials to a message, insert a timestamp, sign the messages, and encrypt the message. The protocol ensures authentication with security tokens. Security tokens in combination with XML Encryption ensure confidentiality while security tokens in combination with XML Digital Signatures ensure integrity, of the SOAP messages.

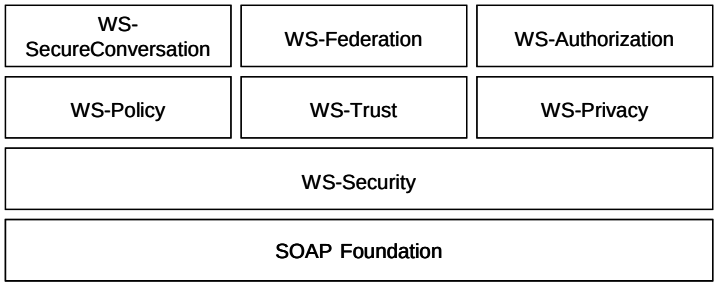

Fig. 9. Web service security specifications.

Apart from WS-Security, web service security specifications also include WS-Policy which defines the rules for service interaction, WS-Trust which defines trust model for secure exchanges and WS-Privacy which states the maintenance of privacy of information. Built with these set of basic specifications are the specifications, WS-SecureConversation that specifies how to establish and maintain secured session for exchanging data, WS-Federation which defines rules of distributed identity and its maintenance, and WS-Authorization which processes the access rights and

exchangeable information. The set of web service security specifications are shown in figure 9. [52]

SAML.

Security Assertion Markup Language (SAML) from OASIS primarily provides single sign on (SSO), cross domain interoperability, means of implementing the basic WS-Security standard through assertions, and helps in managing identity control across domains and organizations - for enhanced user experience. SAML builds on top of the web service security specifications and provides a means by which security assertions can be exchanged between different service entity endpoints.

The basic components of interest in SAML are assertions, protocols, bindings and profiles. SAML assertions carry the authentication information while SAML request/response protocols tell how and what assertions can be requested. Bindings define the transportation of SAML protocols over SOAP/HTTP protocol. A SAML profile can be created using the bindings, protocols along with the assertion structure. The SAML request or SAML response will reside in SOAP body.

SAML request/response protocol binding over SOAP will provide assertions in the SOAP body with information about authentication and authorization. Then SAML assertions are used along with the WS-Security element which will reside in SOAP header. As the SAML assertions contain key of the holder, it can be used to digitally sign the SOAP body. At the receiver end, the signature is verified with the help of the key and the access controls within the assertion.

Extensible Access Control Mark-Up Language (XACML) [53] defines syntax and semantics of a language to express and evaluate access control policies. SAML can also be used independently with other access control mechanisms. When both SAML and XACML are used together, they result in two additional components: Policy Enforcement Point (PEP) and Policy Decision Point (PDP). When PEP receives requests from requestor, it accesses assertions from the requestor and extracts other typical information such as time of request, location etc. and sends it to PDP. PDP then evaluates the request by obtaining related policies and passes on the decision to PEP which enforces the decision towards the requestor.

LA.

Liberty Alliance project [54] is the only global body which is working to define and provide technology, knowledge and certifications to build identity into the foundations of mobile and web service communication. It mainly concentrated on federated identity, because of the lack of connectivity between identities for internet applications in the current wireless technology especially in mobile networks.

The basic components of Liberty Alliance are principal, identity provider and service provider. Principal is the requestor who needs to be authenticated. Identity provider is the one which authenticates and asserts the principal’s identity. The basic provisions of this project are federation which establishes relationship between any two of the above mentioned components, Single Sign On (SSO) where the authentication provided to principal by the identity provider can be maintained to other components such as service providers, and circle of trust where trust will be established between service providers and identity providers with agreements upon

which principals can make transactions and exchange information in a seamless and secure way.

OMA.
Open Mobile Alliance (OMA) group is concentrating to have a unique specification/framework for mobile data services to achieve interoperability. OMA was formed in June 2002 by nearly 200 companies including the world's leading mobile operators, device and network suppliers, information technology companies and content and service providers. Mobility and roaming are the obvious key characteristics which are hindrances to mobile web service interactions [8]. The current possible mobile web service applications have a number of drawbacks as following. First, the applications should be created through tightly-coupled, costly and close alliances between value-added service providers. Second, they have to be created based on a mixture of mostly propriety models and disparate standards such as WAP, Location, Presence, Identity etc. Furthermore, most of the standards to develop these applications have been devised specifically for the mobile environment from the ground up. All these drawbacks will draw high complexity to deploy, integrate and use these applications and services.

The OMA Web Services Enabler specification [55] is destined to cover all the drawbacks mentioned above and envisioned to support the following mobile web service interactions:

- Server-to-server
- Server-to-mobile terminal
- Mobile terminal-to-server
- Mobile terminal-to-mobile terminal (peer-to-peer)

3.2 Security Realization and Analysis

As discussed earlier, secure provisioning of mobile web services needs proper message-level security consisting data integrity, confidentiality and end-point access security that constitutes authentication and authorization. Since, there exists no approved specific mobile web service standards and lot of propriety interfaces are involved, the security was analyzed on a case-by-case scenario.

Security Analysis Design Model.
To secure the communication of our mobile web services provisioning, first we have analyzed the adaptability of WS-Security in the mobile web services domain [17]. The WS-Security adds many performance overheads to the mobile web service invocation cycle. Mainly, extra CPU capabilities are required to process the WS-Security related header elements. The transportation delays also increase significantly as the SOAP message size increases with the added security headers. Figure 10 depicts our architecture to analyze the basic security principles for the Mobile Host.

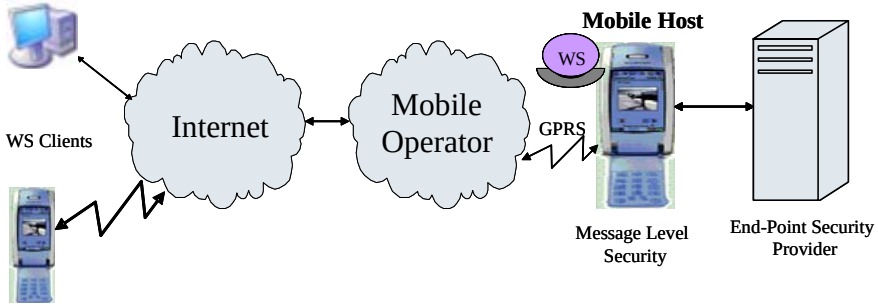

Fig. 10. Proposed security realization scenario of Mobile Host.

The Mobile Host was developed and deployed on a smart phone. Once a web service is deployed on Mobile Host; any web service client can request for the service. The SOAP message along with the WS-Security information is routed across the Internet and mobile operator proprietary network to the Mobile Host. The message-level security information is extracted and addressed at the Mobile Host while the end-point access security is handled by a third party on behalf of the Mobile Host. Then the corresponding service details are extracted and the service is invoked. The SOAP response is sent back to the client across the same route. The performance of the Mobile Host and the network latency were observed while processing the client request.

For providing proper end-point security for the mobile web service provisioning, the basic service-level authentication and user-intervened authorization were realized. In the service-level authentication, an authentication service is provided at the Mobile Host which accepts a username and password and validates the client. Authentication can be password based, Public Key Infrastructure (PKI) based or certificate based. We have considered password based over PKI based authentication due to platform restrictions. The Mobile Host stored the authentication details at the smart phone itself. This posed further problems with the resource constraints of the smart phone, as the authentication information needed extra resources. An alternative for this scenario is provided, where the Mobile Host generates an authentication request to a standalone web service, deployed on an Axis [43] based web service provider (End-point security provider), on behalf of the client, using the authentication information provided by the client. The client can then access any service provided by the Mobile Host. Both the authorization-service request and the service request must be generated in a single session. An alternative for the authentication would be the single sign on addressed by SAML and LA specifications.

In the user-intervened authorization, each of the services provided at the Mobile Host can be configured to obtain the providers (person using the Mobile Host) acceptance before providing the respective service to the web service requestor. Critical issues like disapprovals, user being busy and timeouts were also considered. The process was also automated by using an access control mechanism, based on the authentication details. Realization of single sign on, where identity and credentials can be maintained for multiple sessions and parties is currently under study and our

future publications will address this issue. The following sections provide our analysis and results with message-level security for mobile web services.

The message-level security was further broken-down and the performance penalties of different encryption and signing algorithms were analyzed at the Mobile Host, individually. The breakdown was required to observe the best possible scenario for securing mobile web services communication. This has left us with four different test cases for the analysis of message-level security for mobile web services.

- Unsecured mobile web service communication
- Encrypted mobile web service communication
- Signed mobile web service communication
- Encrypted and Signed mobile web service communication

Security Analysis Implementation Model.

To analyse the WS-Security for Mobile Host, we have used two Sony Ericsson P910i smart phones as web service requestor and the Mobile Host. The smart phones had an internal memory of 64 Kb and ARM9 processor clocked at 156MHz. The phones were connected to the Internet using a GPRS connection. The Mobile Host was rebuilt using J2ME for the security analysis. The P910i device supports MIDP2.0 [56] with CLDC1.0 [57] configuration. For cryptographic algorithms and digital signers, Java based light weight cryptographic API from Bouncy Castle crypto package [58] is used. KSOAP2, the Java API based on KXml2, is adapted according to WS-Security standard and utilized to create the request/response web service messages.

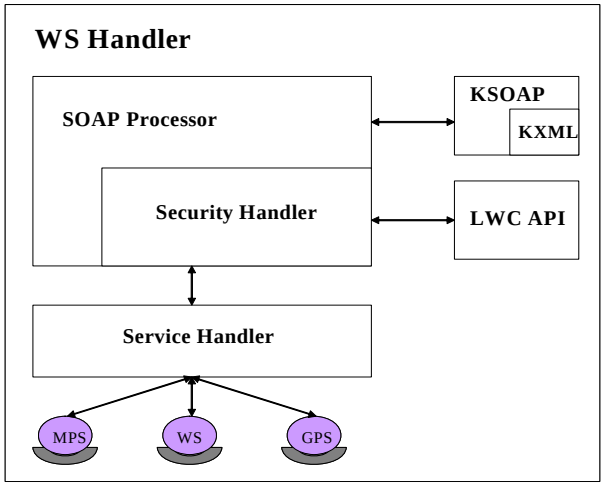

Fig. 11. Web Service Handler of the Mobile Host

The web service security enabled WS Handler component of the Mobile Host is shown in figure 11. The WS Handler receives the web service messages from the HTTP interface of the Mobile Host. The SOAP Processor extracts the SOAP messages from web service requests. The security handler does the respective security tasks/checks over the message and transfers decrypted message to the service handler,

which extracts the service details and invokes the respective service. Effectively, the WS Handler manages the full message-level security and assists in end-point security.

To analyze confidentiality, the message was ciphered with symmetric encryption algorithm and the generated symmetric key is exchanged by means of asymmetric encryption method. The message was tested against various symmetric encryption algorithms [59] including the WS-Security mandatory algorithms, namely, TRIPLEDES [60], AES-128, AES-192 and AES-256 [61]. The PKI algorithm used for key exchange was RSA-V1.5 [62] with 1024 and 2048 bit keys. Upon successful analysis of confidentiality, we tried to ensure data integrity of the messages. The messages were digitally signed and were evaluated against two signature algorithms, DSAwithSHA1 (DSS) [63] and RSAwithSHA1 with 1024 and 2048 bit keys. The effect of signing on top of encryption was also studied later, considering the best algorithms from the individual analysis of encryption and signing. Note that, as said earlier, all the algorithms mentioned above have been implemented using Java based light weight bouncy castle cryptographic API.

All of the above test cases were observed with different message sizes. The size of the request message was 1 Kb while the size of the response messages ranged from 1-10Kb. All the experiments were repeated at least 5 times and the mean of the values were observed for drawing conclusions, to have statistically valid results.

Performance Model of the Mobile Host.

To analyze the performance of the Mobile Host with the security load, the durations of different activities across the mobile web service invocation cycle are observed. The client initiates the call for the web service and the Mobile Host processes the request, populates the response, and sends response back to the client. Similar model was primarily considered for analysing the performance of Mobile Host with out the security incorporation. The model is extended and explained here to contain the security parameters as well.

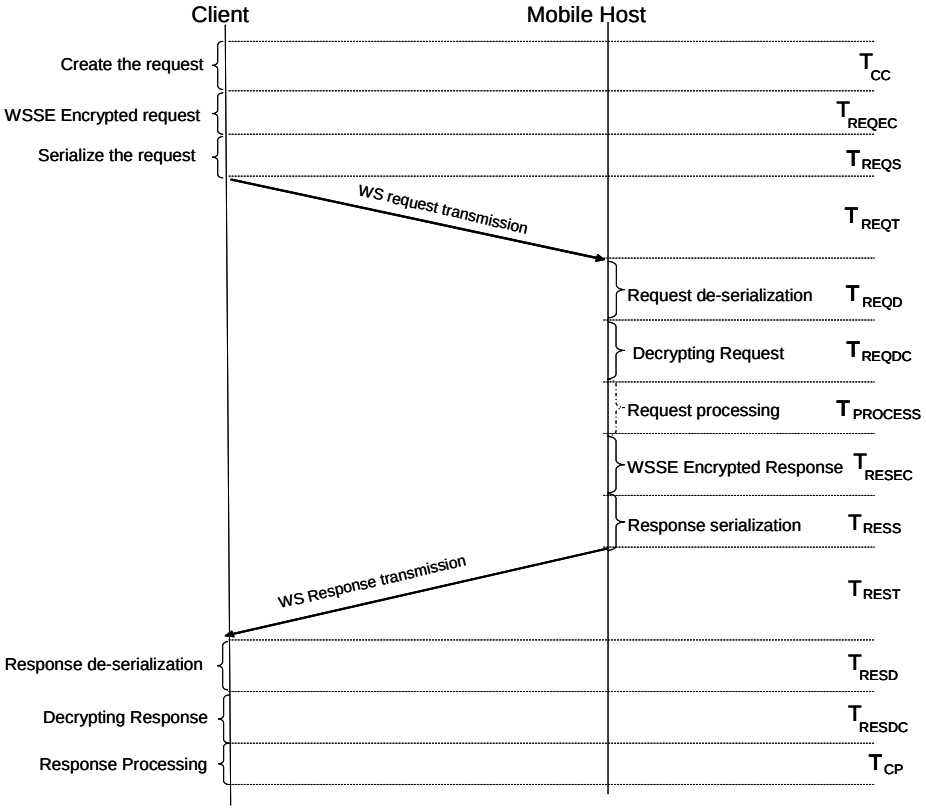

Fig. 12. Secured mobile web service invocation: operations and time stamps.

The total time taken for this mobile web service invocation (Tmwsp) constitutes, the time taken by client for constructing valid SOAP message (Tcc), the time taken to encrypt the message with security information according to WS-Security standard (Treqec), the time taken to serialize the encrypted message (Treqs), the time taken to transmit the SOAP request to Mobile Host (Treqt), the time taken for de-serializing the XML based SOAP request message (Treqd), the time taken to decrypt the request message (Treqdc), the time taken by the Mobile Host to execute the respective business logic and to populate the response (Tprocess), the time taken to encrypt the response message with security information (Tresec), the time taken for serializing the encrypted response message back to XML data streams (Tress), the time taken to transmit the SOAP response back to the client (Trest), the time taken to de-serialize the response at the client (Tresd), the time taken by the client to decrypt the response message (Tresdc), and lastly the time taken by the client to process the response (Tcp). The invocation process is shown in Figure 12 and the total time taken for the mobile web service invocation is given in equation 1.

$$Tmwsp = Tcc + Treqec + Treqs + Treqt + Treqd + Treqdc + Tprocess + Tresec + Tress + Trest + Tresd + Tresdc + Tcp \quad (1)$$

The exact estimation of the Treqt and Trest time is not possible as their calculation process needs the synchronization of time stamps of both Mobile Host and client. Moreover these transmission times were observed during our previous analysis [14]. Those results showed 90% of total invocation cycle is transmission time. So to analyze the minute extra delays due to security load, the whole invocation cycle is observed with both the invocation and processing of the web service request at the Mobile Host itself, thus eliminating the transmission aspects.

Evaluation of the Security for Mobile Host.

The main idea of our study was to realize the WS-Security standards for the Mobile Host. For achieving this, different encryption algorithms, signer algorithms and authentication principles were analyzed in the mobile web service provisioning domain. The performance of the Mobile Host was observed for reasonable quality of service. The parameters of interest were extra delay and variation in stability of the Mobile Host with the introduction of the security overhead. Some of the results are discussed here.

To analyze the effects of message-level encryption on the mobile web service invocation cycle, the messages were encrypted with IDEA [64] with 128 and 256 bit keys, DES [65] with 64 and 192 bit keys and AES with 128, 192 and 256 bit keys. The keys were exchanged using RSA with key sizes 1024 and 2048 bits. Figure 13 summaries the results of our encryption analysis and shows the comparison of latencies for different encryption algorithms with keys exchanged using RSA 1024.

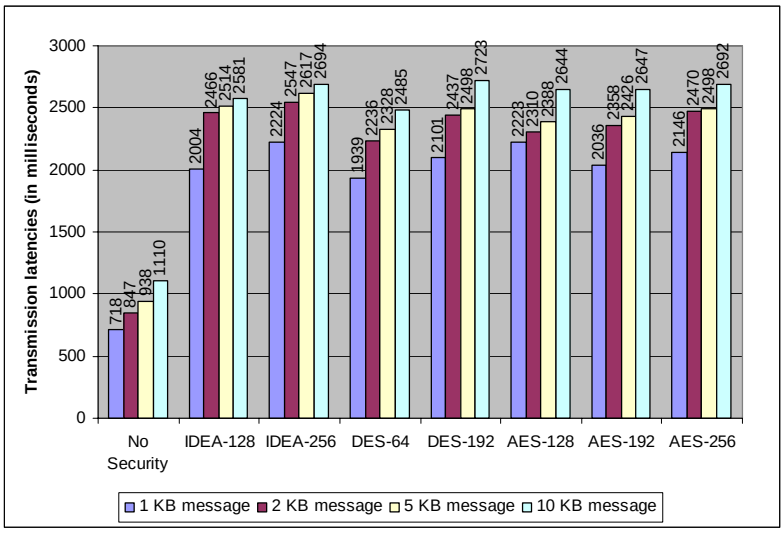

Fig. 13. Performance latencies with various symmetric key encryption algorithms and exchanging keys with RSA 1024.

The results suggest that AES 192 encryption turns out be the best symmetric key encryption method. But the difference in latencies with AES 192 and AES 256 are not so significant. So the best means of encrypting the message would be to use AES 256

bit key and to exchange the message with RSA 1024 bit key, both in terms of provided security and performance penalty. Still the increased latency with this best scenario is approximately 3 times the latency without any security. The extra delays mainly constitutes the times taken for encryption of the request at the client (Treqec), the decryption of the request at the Mobile Host (Treqdc), the encryption of the response at the Mobile Host (Tresec) and the decryption of the response at the client (Tresdc). From the performance model, we can derive this mobile web service message security effort (Tmwsse) as follows:

$$Tmwsse \sim= Treqec + Treqdc + Tresec + Tresdc \quad (\mathbf{2})$$

To analyze the effects of signing on the mobile web service invocation cycle, the messages were signed with two digital signature algorithms, DSAwithSHA1 (DSS) and RSAwithSHA1 with 1024 and 2048 bit key sizes. The results suggested that the best way to sign the mobile web service message would be using RSA V1.5 with 1024 bit key. RSA algorithm is preferred ahead of the DSA, considering the performance latencies.

With a key size greater than 1024, both key exchange and signing were observed to be beyond the resource capabilities of smart phones. It was also observed that the latency caused by signing is slightly higher than the latency caused by the encryption, especially, when considering DSS signature.

After successful analysis of encryption and signing, we have analyzed the performance of signing on top of message-level encryption. Figure 14 depicts times taken for various phases of a message-level secured web service request/response cycle. The timestamps does not include the transmission delays. The transmission delays were deliberately eliminated, to be able to observe the minute timestamps of the remaining activities in the mobile web service invocation cycle. The original message was ciphered with AES-256 algorithm and its key is exchanged with RSA-1024 PKI algorithm. To summarize further, the request message size was 1 Kb and response message size was 2 Kb. The total cycle for highly secured communication, AES-256 bit ciphered, cost around ~3 sec with RSAwithSHA1 signature and ~5.5 sec for DSAwithSHA1 signature. The comparison of mobile web service invocation cycle time stamps for messages signed with RSAwithSHA1 and DSAwithSHA1 are shown in figure 15.

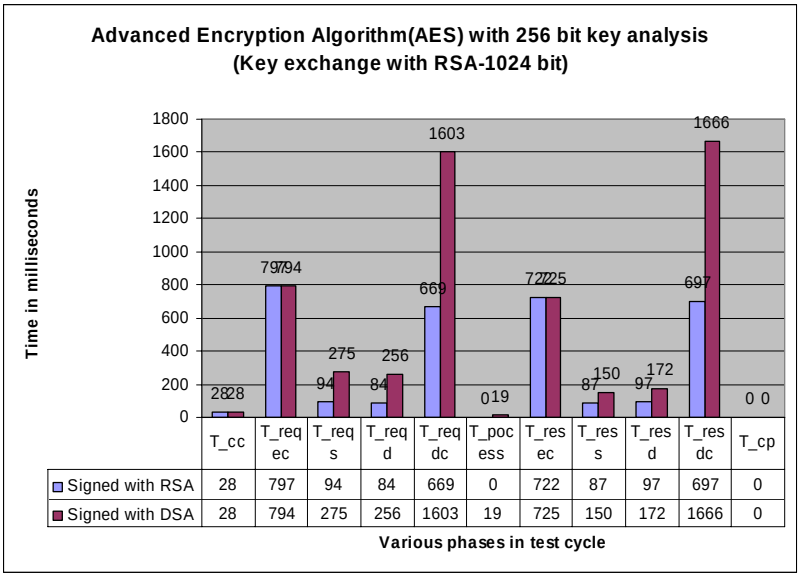

Fig. 14. Latencies of various phases of a message-level secured mobile web service invocation cycle.

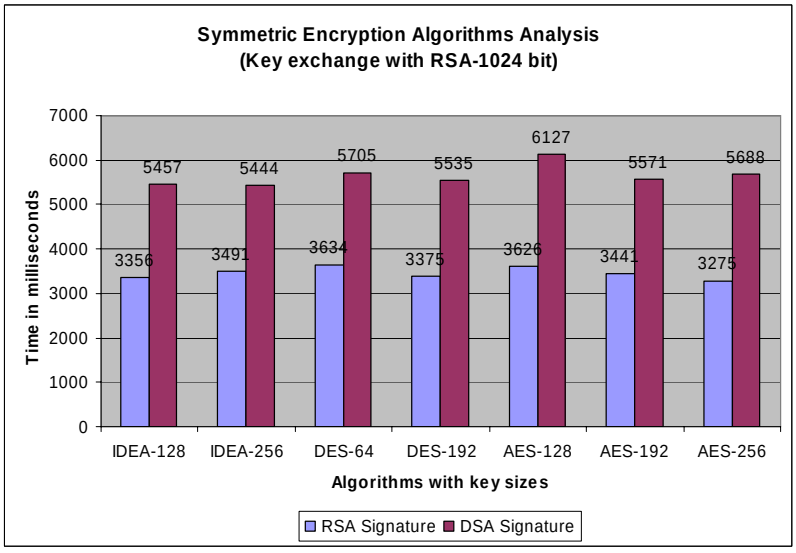

Fig. 15. Comparison of timestamps of web service invocation cycle with various symmetric key encryption algorithms.

From the analysis shown in figure 14 and figure 15, we can conclude that the best way of securing messages in mobile web service provisioning is to use AES

symmetric encryption with 256 bit key, and to exchange the keys with RSA 1024 bit asymmetric key exchange mechanism and signing the messages with RSAwithSHA1. But there are still high performance penalties when the messages are both encrypted and signed. So we suggest encrypting only the parts of the message, which are critical in terms of security and signing the message. The signing on top of the encryption can completely be avoided in specific applications with lower security requirements.

Effects of WS-Security on Size of the Message.
The increase in size of the message with the security headers is also quite daunting. A typical web service message after applying the WS-Security is shown in figure 16. The SOAP message body can be completely encrypted or only parts of the message can be encrypted. The ciphered data is stored in the body of the updated message. The security information like encryption algorithms used, keys, digests, signing information is maintained in the SOAP header. The message shown below is the snapshot of a message encrypted with AES, and the key exchanged with RSA V 1.5. The message was later signed with RSAwithSHA1.

```
<v:Envelope ...>
 <v:Header>
  <Security>
   <n1:EncryptedKey ...>
    <EncryptedMethod Algorithm="...#rsa-1_5" />
    <CipherData>
     <CipherValue>...</CipherValue>
    </CipherData>
    <ReferenceList>
       <DataReference URI="#4412525"/>
     </ReferenceList>
   </n1:EncryptedKey>
   <n2:Signature ...>
    <SignedInfo>
     <SignatureMethod Algorithm="...#rsa-sha1" />
     <Reference>
      <DigestMethod Algorithm="...#sha1" />
      <DigestValue>...</DigestValue>
     </Reference>
    </SignedInfo>
    <SignatureValue>...</SignatureValue>
    <KeyInfo>
     <KeyValue>
      <RSAKeyValue>
       <Modulus>...</Modulus>
       <Exponent>AQAB</Exponent>
      </RSAKeyValue>
     </KeyValue>
    </KeyInfo>
   </n2:Signature>
  </Security>
 </v:Header>
 <v:Body>
  <n0:EncryptedData Id="223940028" ...>
```

```
   <EncryptionMethod Algorithm="...#AESEngine" />
    <CipherData>
     <CipherValue>Ye/qF7...</CipherValue>
    </CipherData>
   </n0:EncryptedData>
 </v:Body>
</v:Envelope>
```

Fig. 16. A typical SOAP message incorporated with WS-Security.

The example showed in figure 16 also hints the increase in size of the message with the added security header information. With our analysis we have observed that there is a linear increase in the size of the message with the security incorporation. The latency in the encrypted message size for a typical 5 Kb message is approximately 50% [66].

3.3 Scalability Aspects of the Mobile Host

Web services communication is a layered communication and across different protocols. Considering SOAP over HTTP, at the lowest level is the transportation protocol, TCP. On top of TCP lies the HTTP communication. Then SOAP communication is over the HTTP protocol. The application communication and protocols for example WS-Security lies on top of SOAP. So any message exchanged over the web service communication, consists some overhead across all the different layers. Since we are considering wireless environments, and the message exchange is over the cellular network, the size of the message has to be reduced to the minimum possible level [67]. So from this discussion, the size of the message

$$Bmsg = Btp + Bmtp + Bsoap + Bapp \quad (3)$$

Where Btp, Bmtp, Bsoap, Bapp are the message overheads over transportation, message transportation, SOAP, application protocols respectively. So to exchange the messages effectively the Bmsg has to be minimized. So let us consider shedding some overhead at each level. We also should note that the minimal encoding may not always be the best solution. First reason for this is that the encoding should be efficient, both in terms of size reduced and extra performance penalties added. For example if the size of message is reduced by 50% and the processing of the encoding takes more than half the time of actual message exchange cycle, the encoding mechanism is not efficient. Secondly the encoding mechanism should not affect the interoperability.

So if an attempt is made to reduce the overload at Btp or Bmtp, the interoperability of the web services is seriously impeded. So the best position to target the encoding process is at the Bsoap level. So the XML based SOAP messages are to be compressed. We are currently focusing at different XML compression [68], [69] and SOAP optimization techniques [70], to reduce the size of the message to be transmitted, there by improving the scalability of the Mobile Host. The compression of the web service messages can in turn reduce the security load, as the content to be encrypted or signed becomes less with compression, and thus improves the performance of the Mobile Host. But only in terms of security processing there is not

significant difference with increase in size, at least until 10 Kb message sizes, as from our results shown in figure 13.

From this QoS analysis we can conclude that the results of our security study are welcoming and the mobile web service messages of reasonable size, approximately 2-5kb, can be secured with web service security standard specifications. But based on our till-date realization of security awareness in cellular networks, we conclude that secure web service provisioning in mobile networks is still a great challenge. The mechanisms developed for traditional networks are not always appropriate for the mobile web services environment and this area still holds ample room for further research. From the analysis we also recommend some support at the hardware level, handling the security details for the smart phones. But having a hardware support might effect the interoperability, and the hardware need to be upgraded with each software update at the security algorithm level. A trade-off between the two arguments is yet to be met at this level.

Our future research in this domain includes providing proper end-point security for the Mobile Host with federated identity and appropriate single sign on strategy, using SAML and LA standards. We also want to have a detailed performance analysis of the Mobile Host with full security features through real-time applications. We are also looking for alternatives, to reduce the security and scalability processing load on the Mobile Host. We are trying to realize an Enterprise Service Bus (ESB) [71] based Mobile Web Services Mediation Framework (MWSMF) [72], which maintains the individual user profiles, personalization settings and context sensitive information. The MWSMF helps in maintaining the QoS of the Mobile Host. With the mediation framework in place the communication between the client and the middleware could be based on WS-Security and the communication between the middleware and the Mobile Host be based on a security mechanism feasible for mobile web services. The transformation of the messages between the two standards will also be handled at the mediation framework. Similar transformations can be maintained at the mediation framework handling the scalability issues. In the scalability scenario the communication between the client and the middleware is based on the basic web services standard and the communication between the middleware and the Mobile Host be based on a technique that compresses web service message exchange.

4 Mobile Web Service Discovery

We have discussed many applications with the Mobile Host, in the previous sections. While the applications possible with mobile web services are quite welcoming, the huge number of web services possible, with each Mobile Host providing some services in the wireless network, makes the discovery of these services quite complex. Proper discovery mechanisms are required for successful adoption of mobile web services into commercial environments. The traditional centralized UDDI based registries [18] have many limitations in this aspect and might not be the perfect solution for the mobile web service discovery. The dynamic nature of the mobile nodes further enhances this problem.

4.1 Discovery Aspects of Mobile Web Services

Generally, in standard web services, once a service provider develops and deploys the service, he publishes the service with a UDDI registry. The registry maintains a reference of the WSDL documents. The WSDL document, that defines and describes a web service, consists of information specific to the location of the service (binding information) and the operations (methods) the service exposes. Any potential web service client searches for the service in the public registry, gets the description of the service and tries to access the service using the information specified by the WSDL. Similar to web service invocation, the communication between client and UDDI registry is also based on SOAP. Since the Mobile Host is implemented on the smart phone, mostly by using the basic web services architecture, the standard WSDL and UDDI registry can theoretically be used to describe and publish the services. Obtaining the binding information of the mobile web services is tricky as it needs the IP address of the Mobile Host, where the services are deployed.

But in a commercial environment with Mobile Hosts, and with each Mobile Host providing some services in the wireless network, the bulk of services expected to be published could be quite high. In such a situation, a centralized solution is not a best idea, as they can have bottlenecks and can make single points of failure. Besides, mobile networks are quite dynamic due to the node movement. Nodes can join or leave network at any time and can switch from one operator to another operator. This makes the binding information in the WSDL documents, inappropriate. Hence the services are to be republished every time the Mobile Host changes the network or its binding information. This process leaves many stale advertisements in the registry. Keeping up to date information of the published mobile web services in centralized registries is really difficult. So we are studying alternate means of discovering the web services deployed with Mobile Hosts. Here we will be addressing our proposed solution using P2P networks. Before explaining the discovery of mobile web services in P2P network, we discuss the issues with Mobile Host's entry into P2P network.

4.2 Mobile Web Service Provisioning in P2P Networks

During our application analysis of Mobile Host, it was observed that most of the targeted collaborative applications, somehow converged to P2P applications and P2P offered a large scope for many applications with Mobile Host. P2P is a set of distributed computing model systems and applications, used to perform a critical function in a decentralized manner. Peers are autonomous and in its pure form; each peer acts as both server and client. P2P takes advantage of resources of individual peers like storage space, processing power, content and achieves scalability, cost sharing and anonymity, and thereby enables ad-hoc communication and collaboration. P2P systems have evolved across time and have wide range of applications and provide a good platform for many data and compute intensive applications [73], [74].

The first generation P2P systems like Napster [75] used centralized servers for maintaining an index of the connected peers and their resources. The indexes can later be queried by the peers and the resources are downloaded from the provider peers using IP networks. But these centralized systems have single points of failure and

produce giant communication traffic and storage on the server resulting bottlenecks. These drawbacks lead to the second generation of P2P systems like Gnutella [73] which used a complete decentralized network. Unlike Napster, Gnutella would connect users directly to a group of other users and so on. For this, Gnutella uses pre-existing, extendable list of possible working peers, whose addresses are embedded inside the application code. But these decentralized networks formed islands in the P2P network and their search functions were unreliable and may not query entire network. The third generation P2P systems like eDonkey [76] and Bit Torrent [77] are a hybrid of the previous two generation technologies and made enhancements to improve their ability to deal with large numbers of users using concepts like super peers. Super peers have higher resource capabilities and act as relays for other peers and super peers. Super peers also have abilities to traverse NAT and firewall.

Analogous to web services, P2P systems can also leverage SOA and are also designed to enable loosely coupled systems. The concept of services and the similarities of description stack of both P2P and web services make them comparable [78]. The major difference being; web services will be well-known hosts with static IP addresses, and are based on a centralized model and primarily focused on standardizing messaging formats and communication protocols. P2P systems, on the other hand, are based on a decentralized model and primarily focused on supplying processing power, content, or applications to peers in a distributed manner, and less focused on the semantics of messaging formats and communication protocols. In the P2P world the peers jump through potential jumble of firewalls, NATs and proxies trying to connect to other peers.

In order to reap the benefits of P2P, by achieving increased application scope, and targeting efficient utilization of resources of individual mobile peers, we are trying to adapt Mobile Host into P2P networks. For this many of the current P2P technologies like Gnutella, Napster and Magi [79] are studied in detail. Most of these technologies are proprietary and are generally targeting specific applications. Only project JXTA [20] offers a language agnostic and platform neutral system for P2P computing. JXTA technology is a set of open protocols that allow any connected device on the network ranging from cellular phones and wireless PDAs to PCs and servers to communicate and collaborate in a P2P manner. JXTA enables these devices running on various platforms not only to share data with each other, but also to use functions of their respective peers. JXTA peers use XML as standard message format and create a virtual P2P network over these devices connected over different networks.

Moreover the JXTA community has developed a light version of JXTA for mobile devices, called JXME (JXTA for J2ME) [80]. JXME works on MIDP supporting devices like smart phones. JXME simplified Mobile Host's entry into P2P domain. JXME has two versions: proxyless and proxied. The proxyless version works similar to native JXTA, whereas the proxied version needs a native JXTA peer to be set up as its proxy. The proxied version is lighter of the two versions and peers using this version participate in binary communication with their proxies. Considering JXTA also eliminates many of the low level details of the P2P systems like the transportation details. The peers can communicate with each other using the best of the many network interfaces supported by the devices like ethernet, WiFi, GPRS etc. Moreover JXTA dynamically uses either TCP or HTTP protocols to traverse network barriers, like NATs and firewalls.

Considering these advantages and features of the JXTA, the Mobile Host was adapted into the JXTA network, to check its feasibility in P2P networks. Figure 17 shows the architecture of final deployment scenario of Mobile Hosts in the JXME network.

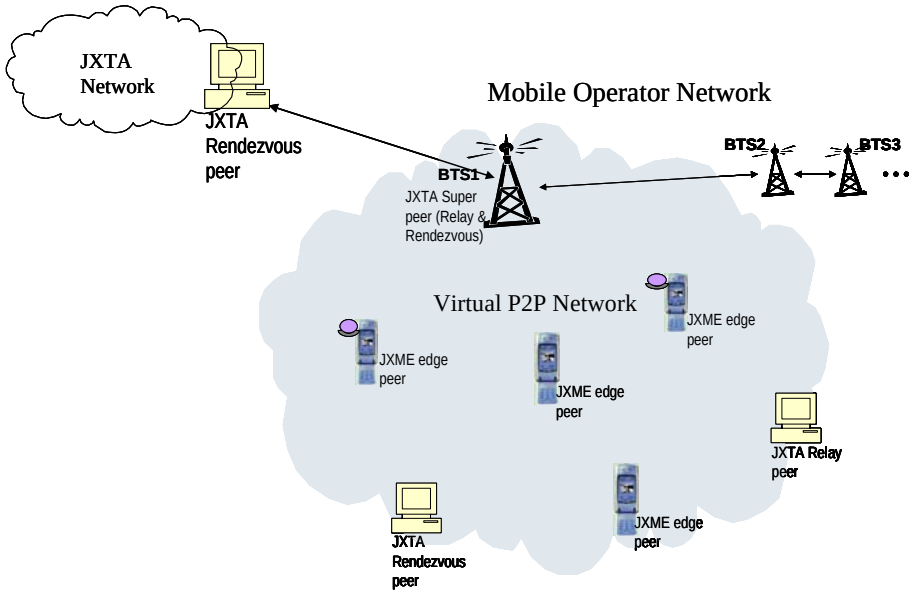

Fig. 17. Virtual mobile P2P network with Mobile Hosts.

As shown in figure 17, the virtual P2P network is established in the mobile operator network with one of the node in operator proprietary network, acting as a JXTA super peer. JXTA network supports different types of peers to be connected to the network. The general peers are called edge peers. An edge peer registers itself with a rendezvous peer to connect to the JXTA network. Rendezvous peers cache and maintain an index of advertisements published by other peers in the P2P network. Rendezvous peers also participate in forwarding the discovery requests across the P2P network. A relay peer maintains route information and routes messages to peers behind the firewalls. A super peer has the functionality of both relay and rendezvous peers. In the mobile P2P network, the super peer can exist at Base Transceiver Station (BTS) and can be connected to other base stations, thus extending the JXTA network into the mobile operator network. Any Mobile Host or mobile web service client in the wireless network can connect to the P2P network using the node at base station as the rendezvous peer. The super peer can also relay requests to and from JXTA network, to the smart phones. Standalone systems can also participate in such a network as both rendezvous and relay peers, if the operator network allows such functionality, further extending the mobile P2P network.

Mobile Host in JXME network offers many advantages in domains like collaborative learning, image sharing, and location based services etc., taking advantage of individual peers' resources like storage space, processing power.

Moreover, the mobile phone users in the operator network might not use the web services for the development purpose. General mobile users are interested in applications rather than individual components or web services. An application might use one or more web services at the backend and can be provided as an installable application. In such a situation, the P2P network can offer easy means of storing and sharing these installable client applications for the participating peers.

Not just the enhanced application scope, the JXME network also provides many technical advantages to the Mobile Host like enhanced service discovery and access mechanisms. With in JXTA network, each peer is uniquely identified by a static peer ID, which allows the peer to be addressed independent of its physical address like DHCP based IP address. This peer ID will stay forever with that device even though the device supports multiple network interfaces like Ethernet, WiFi for connecting to the P2P network. By using peer ID, Mobile Host does not have to worry about changing IPs and operator networks, and is always visible to the web service client. Mapping the peer ID to the IP is taken care by the JXTA network, thus eliminating the need for public IP. The public IP for each of the participating Mobile Hosts was observed to be the major hindrance for commercial success of Mobile Host.

4.3 Discovery of Mobile Web Services in JXTA Network

In JXTA the decentralization is achieved with the advertisements. All resources like peers, peer groups and the services provided by peers in JXTA network are described using advertisements. Advertisements are language-neutral metadata structures represented as XML documents. Peers discover each other, the resources available in the network and the services provided by peers and peer groups, by searching for their corresponding advertisements. Peers may cache any of the discovered advertisements locally. Every advertisement exists with a lifetime that specifies the availability of that resource. Lifetimes gives the opportunity to control out of date resources without the need for any centralized control mechanism. To extend the life time of an advertisement, the advertisements are to be republished.

Thus to achieve alternate discovery mechanism for mobile web services, the services deployed on Mobile Host in the JXTA network are to be published as JXTA advertisements, so that they can be sensed as JXTA services among other peers. JXTA specifies ‘Modules’ as a generic abstraction that allows peers to describe and instantiate any type of implementation of behaviour in the JXTA world. So the mobile web services are published as JXTA modules in the P2P network. The module abstraction includes a module class, module specification, and module implementation. The module class is primarily used to advertise the existence of a behaviour. Each module class contains one or more module specifications, which contain all the information necessary to access or invoke the module. The module implementation is the implementation of a given specification. There might be more than one implementation for a given specification across different platforms. Figure 18 shows the mapping between JXTA modules and web services. The collection of module abstractions represent the UDDI in a sense of publishing and finding service description and WSDL in a sense of defining transport binding to the service.

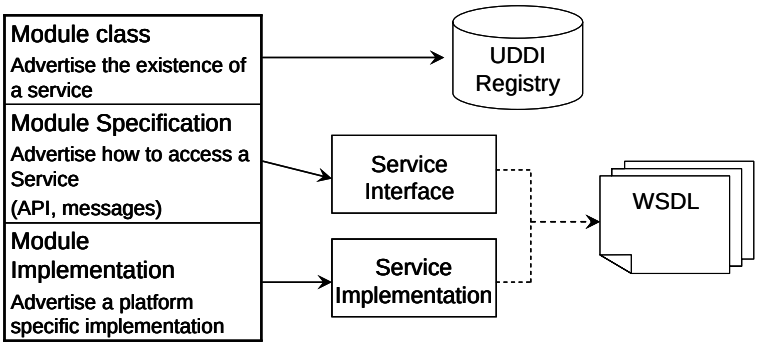

Fig. 18. Mapping between JXTA modules and Web Services.

To publish the mobile web services in the JXTA network, a standard Module Class Advertisement (MCA) is published into the P2P network, declaring the availability of a set of web service definitions, in that peer group. Once new web services are developed for the Mobile Host, the WSDL descriptions of these services are incorporated into the Module Specification Advertisements (MSA), and are published into the P2P network. The MSAs are published into JXME network with an approximate life time that specifies the amount of time the Mobile Host wants to provide the service. The MSAs are cached at rendezvous peers or any other peers, with sufficient resource capabilities. Once the life time expires the MSAs are automatically deleted from the P2P network, thus avoiding the stale advertisements. If the Mobile Host wants to extend the life time of the provided service, the MSA can be republished. The MSA can be published into the network by a service developer or even by the Mobile Host. The structure of the MSA is shown in figure 19.

```
<?xml version=”1.0” encoding=”UTF-8”?>
<jxta:MSA>
<MSID> . . . </MSID>
<Name> . . . </Name>
<Crtr> . . . </Crtr>
<SURI> . . . </SURI>
<Vers> . . . </Vers>
<Desc> . . . </Desc>
<Parm>
    <WSDL>
          <definitions …>
                <message …>  . . . </message>
                <portType …>  . . . </portType>
                  . . .
          </definitions>
    <WSDL>
</Parm>
<jxta:PipeAdvertisement> . . . </jxta:PipeAdvertisement>
<Proxy> . . . </Proxy>
<Auth> . . . </Auth>
</jxta:MSA>
```

Fig. 19. Structure of module specification advertisement (MSA) advertising a web service.

The MSA contains unique identifier (MSID) that also includes the static module class ID, which identifies the web services module class advertisement. The other elements of MSID include name, creator, specification and description of the advertisement. The optional element Parm consists the description (WSDL) of the web service being advertised. The PipeAdvertisement consists the advertisement of the pipe which can be used to connect to the specific web service deployed on the Mobile Host. The receiving endpoint of the pipe can be addressed with a Peer ID of the respective peer. Thus if the invocation of mobile web service is across the JXTA network, using pipes, the need for public IP is eliminated. This sort of invocation is being studied and for the time being the Mobile Host is addressed with IP and once the web services are discovered the communication between the Mobile Host and mobile web service client is still SOAP over HTTP. The remaining two elements, Proxy and Auth from MSA carry the proxy module and the security (authentication) information of the wed service module.

The module specification advertisements carrying the web service descriptions can be searched by name and description parameters. The JXTA API provides a simple keyword search on the name and description elements of the modules advertised in mobile P2P network. As we are considering about huge numbers of mobile web services, these basic parameters might not be sufficient to find out the exact search results. In fact, some valuable information like context information may not be included in these basic XML tags. Moreover we would like to extend the search criteria to the WSDL level. This means that search parameters would not be restricted to module specification advertisement details. The search will also extend by looking up the WSDL tags and information. The main idea behind this approach is that people usually express their opinion by using frequently used words and the frequency of a keyword in WSDL description is also relevant. To handle this, advanced discovery of mobile web services in P2P, index searching tools are used to match the best suited services.

This detailed search mechanism can not be performed at the JXME edge peer because of the resource limitations of the smart phones. So, the advanced search mechanism can be shifted to the standalone distributed middleware, MWSMF, just like the QoS components. In the scenario where the Mobile Host uses the proxied version of JXME, the proxy node can be a participant in the mediation framework, handling the discovery issues for the smart phones.

4.4 Advanced Matching/Filtering of Services

As already discussed, the basic mobile web service discovery in JXTA networks, across module specification advertisements is purely based on text based keywords. Hence the search returns a large number of resulted services, returning every service that matches the keyword. Since the discovery client in this scenario is a smart phone, the result set should be quite small so that the user can scroll through the list and can select the intended service. Subsequently to order the JXTA search resulted services according to their relevancy, Apache Lucene tool [81] is used. Lucene is an open

source project hosted by Apache and provides a Java based high-performance, full-featured text search engine library. Lucene allows to add indexing and searching capabilities to user applications. Lucene can index and make searchable any data that can be converted to a textual format. Using the tool and its index mechanism the search results were ordered/filtered and the advance matched services were returned to the discovery client.

Modules advertising the web services in JXTA can also be properly categorized using peer groups. Web services of the same category like services of same publisher, same business type can thus be published in the same peer groups. Hierarchies of peer groups can be maintained in JXTA. Categories help in identification or classification of all the web service types and help in easy discovery of web services. The peer groups thus simulate the tModel feature of the UDDI. Currently we are studying to compare the P2P discovery approach with UDDI and are trying to join the best features of UDDI into our 'mobile P2P discovery' approach.

Once the mobile web services were discovered from the P2P network, the mobile user can scroll through the list of services and can select the best possible service. The web service invocation client is dynamically generated at the mediation framework using the WSDL2Java tools [10], [82] and is downloaded and installed into the smart phone. The deployed client software can then be used to access the service from the Mobile Host. Alternatively mobile applications were developed by composing multiple services and the applications were advertised into the P2P network and were shared using P2P file sharing mechanisms.

4.5 Evaluation of Mobile Web Service Discovery

To evaluate the mobile P2P discovery approach, a JXTA P2P network is established with smart phones connecting to a stand alone relay peer. The relay also acts as a JXME proxy for the mobile phones and thus connecting them to the JXTA network. The relay peer is connected to a stand alone PC, which acts as a rendezvous peer. The rendezvous peer can further connect to other rendezvous peers. Thus the P2P network is established and the network is extended to public JXTA network. The JXME P2P scenario is shown in figure 20.

The mobile web services developed for the smart phones are deployed on the P910i based Mobile Hosts and the services are advertised according to the mobile P2P discovery approach at the rendezvous peer1. Later alternate smart phones are connected to the P2P network using the relay peer, shown in figure 20, searched for the services in the P2P network. The smart phones are successful in identifying the services in the P2P network, with reasonable performance penalties for the Mobile Host. The discovery process took less than a second for most of the services. The scalability of the approach is yet to be verified once the UDDI mapping is finalized.

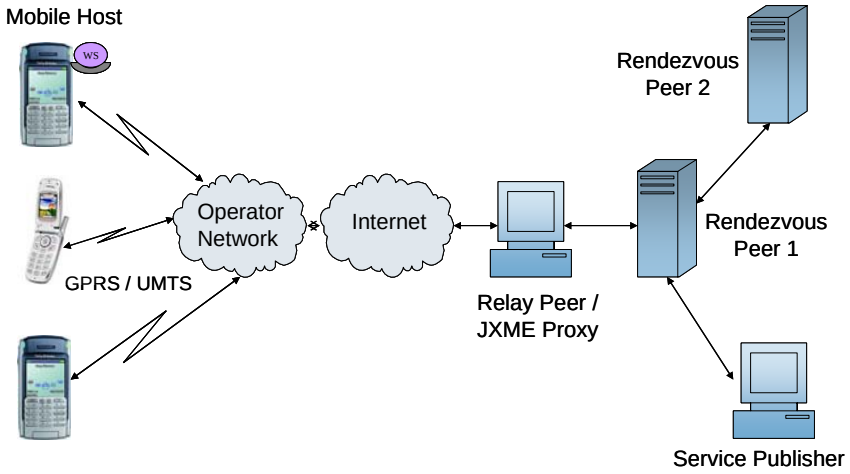

Fig. 20. The P2P based mobile web service discovery evaluation scenario.

Since the mobile web service discovery approach is a keyword based search and the search also extended to WSDL parameters, the relevancy of the resulting services were observed to be a little indistinguishable. Mobile web service clients generally prefer using services of the Mobile Host based on several context parameters such as location, time, device capabilities, profiles, and load on the Mobile Host etc. Most of these details can not be provided just based on keywords. Once the P2P discovery approach finds its way in to the real-time environment, with each Mobile Host providing some services, providing the context information like user profiles and device capabilities is crucial in achieving much valid results.

Semantic matching of services gives the most appropriate and relevant results for mobile web service discovery. The service context and device profiles can be described using ontology-based mechanism. For describing the semantics of services, the latest research in service-oriented computing recommends the use of Web Ontology Language (OWL) [83] based Web Ontology Language for Services (OWL-S) [84]. OWL-S is an ongoing effort to enable automatic discovery, invocation, and composition of web services. Currently we are studying to realize this semantic mobile web service discovery process in P2P networks.

But the semantic discovery process is heavy, in terms of both resource consumption and performance latencies like extra delay. So after the analysis of our approach, we suggest using the P2P discovery mechanism first to reduce the search space. The resulted services can then be matched semantically for the most relevant results. Just as a hint, in terms of numbers, the advanced matching of services should return a set of approximately 50 matched services, of which the semantic matching should reduce the resulted services to a scrollable set (5 - 10) for the smart phones.

5 Conclusion and Future Research Directions

This paper mainly discussed our mobile web service provisioning project with the concepts, performance analysis and application scope of the Mobile Host. Mobile Host is technically feasible and our performance analysis suggested that many basic services can be provided from the smart phones with reasonable performance penalties. From our performance analysis, as the most important result, it turns out that the total web service processing time at the Mobile Host is only a small fraction of the total request-response invocation cycle time (<10%) and rest all being transmission delay, in a GPRS network. This makes the performance of the Mobile Host directly proportional to achievable higher data transmission rates. Thus the higher data transmission rates possible with emerging 3G and 4G technologies make the Mobile Host soon realizable in commercial environments. Mobile Host offers many applications in different domains and some of the scenarios are addressed here.

The paper later discussed the QoS challenges for the Mobile Host, both in terms of security and scalability issues, and tried to adapt some of the existing QoS standards to the mobile web services domain. In terms of security analysis, first we have discussed mobile Host's problems and aspects with security and tried to give the best possible message-level security scenario for mobile web services. The results of our security analysis are welcoming and the mobile web service messages of reasonable size, approximately 2-5kb, can be secured with web service security standard specifications. But based on our till-date realization of security awareness in cellular networks, we conclude that secure web service provisioning in mobile networks is still a great challenge. The mechanisms developed for traditional networks are not always appropriate for the mobile environment and this area still holds ample room for further research. Our future research in the security domain includes providing proper end-point security for the Mobile Host with federated identity and appropriate single sign on strategy, using SAML and LA standards. In terms of scalability, we are focusing at different XML compression and SOAP optimization techniques, to reduce the size of the message to be transmitted, there by improving the scalability of the Mobile Host. The compression of the web service messages can in turn reduce the security load, as the content to be encrypted or signed becomes less, and thus improves the performance of the Mobile Host. But only in terms of security processing at the Mobile Host, there is not significant difference with increase in encrypted message size, at least until 10 Kb message sizes, as from our results shown in figure 13.

In terms of providing proper QoS for Mobile Host, we are looking for alternatives to reduce the security and scalability processing load on the Mobile Host. We are trying to realize an Enterprise Service Bus (ESB) based Mobile Web Services Mediation Framework (MWSMF), which maintains the individual user profiles, personalization settings and context sensitive information. With the mediation framework in place the communication between the web service client and the middleware could be based on WS-* specifications and the communication between the middleware and the Mobile Host be based on the QoS mechanism feasible for mobile web services. The transformation of the messages between the two standards will also be handled at the mediation framework. The communication between mediation framework and the Mobile Host can even be without any QoS parameters,

if the mediation framework can be deployed with the operator network and the operator maintains the QoS terms in the cellular network. If the later scenario is possible, the Mobile Host is completely deprived off the additional QoS loads.

The paper also addressed the concept of publishing and discovery of web services deployed with Mobile Hosts in P2P networks. The approach makes use of the JXTA modules feature and provides an alternative means for discovering the mobile web services. The approach clearly solves the problem of discovering huge number of mobile web services, using resources of individual peers effectively, and at the same time eliminates the problem of inactive (stale) services. The scalability of the approach is yet to be verified and its conceptual mapping with UDDI registry is being studied. We are also interested in extending the approach to the semantic web services domain. Basically we are looking at context aware service discovery considering device, location, and time context and user preferences of the smart phones. Apart from the discovery mechanism, accessing the mobile web service in JXTA network, in addition to their access from the IP network is also of high interest. The access mechanism provides alternative means of addressing the Mobile Hosts and thus eliminates the need for public IP for all participating Mobile Hosts, in an operator proprietary cellular network.

Acknowledgments. This work is supported by German Research Foundation (DFG) as part of the Graduate School ”Software for Mobile Communication Systems” at RWTH Aachen University and partly by the Research Cluster Ultra High-Speed Mobile Information and Communication (UMIC) (http://www.umic.rwth-aachen.de/)

References

1. Box, D. et al.: Simple Object Access Protocol (SOAP), version 1.1., W3C Note, W3C. http://www.w3.org/TR/soap/ (2000)
2. IETF: Hypertext Transfer Protocol version 1.1., IETF RFC 2616. http://www.ietf.org/rfc/rfc2616.txt (1999)
3. Christensen, E., Curbera, F., Meredith, G., Weerawarana, S.: Web Services Description Language (WSDL) 1.1., W3C Working group note. http://www.w3.org/TR/wsdl (2001)
4. Booth, D., Haas, H., McCabe, F.: Web Service Architecture, W3C Working group note. http://www.w3.org/TR/ws-arch/ (2004)
5. Rollman, R., Schneider, J.: Mobile web services, XML 2004 Proceedings by SchemaSof., http://www.idealliance.org/proceedings/xml04/papers/73/MobileWebServices.pdf (2004)
6. 3GPP: Third Generation Partnership Project. http://www.3gpp.org/ (2007)
7. Thomas, K: Fourth Generation (4G) wireless communications. http://www.4g.co.uk/ (1999)
8. OMA: Open mobile alliance overview, Open mobile alliance group. http://www.openmobilealliance.org/docs/OMAShortPaper_May2004v.1.doc (2004)
9. LibertyAlliance: The Liberty Alliance Project. http://www.projectliberty.org/ (2007)
10. Sun: Sun Java Wireless Toolkit. http://java.sun.com/products/sjwtoolkit/ (2007)
11. IBM: WebSphere Studio Device Developer. http://www-306.ibm.com/software/wireless/wsdd/ (2007)
12. Balani, N.: Deliver Web Services to mobile apps, IBM developerWorks, (2003)
13. Srirama, S., Jarke, M., Prinz, W.: Mobile Web Service Provisioning. In: Int. Conf. on Internet and Web Applications and Services (ICIW06). IEEE Computer Society (2006) 120-125

14. Srirama, S., Jarke, M., Prinz, W.: Mobile Host: A feasibility analysis of mobile Web Service provisioning', 4th International Workshop on Ubiquitous Mobile Information and Collaboration Systems, UMICS 2006, a CAiSE'06 workshop. (2006) 942-953
15. Atkinson, B. et al.: Web Services Security (WS-Security), Technical report, Microsoft, IBM and Verisign, April. (2002)
16. Farell, S. et al.: Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML), V1.1., Committee specification, OASIS. (2003)
17. Srirama, S., Jarke, M., Prinz, W. and Pendyala, K.: Security Aware Mobile Web Service Provisioning, In Proceedings of the International Conference for Internet Technology and Secured Transactions, ICITST'06. London, UK, ISBN 0-9546628-2-2, e-Centre for Infonomics. (2006) 48-56
18. UDDI: Universal Description, Discovery, and Integration (UDDI), Technical report, UDDI.ORG. http://www.uddi.org (2000)
19. Shirky, C., Truelove, K., Dornfest, R., Gonze, L., Dougherty, D. (Eds.): P2P networking overview. Sebastopol, CA: O'Reilly. (2001)
20. Gong, L.: JXTA: A network programming environment. IEEE Internet Computing, 5(3). (2001) 88–95
21. Holley, K., Channabasavaiah, K., Tuggle, Jr. E. M.: Migrating to a Service-Oriented Architecture, IBM DeveloperWorks (2003)
22. OMG: Common Object Request Broker Architecture: Core Specification, Object Management Group. http://www.omg.org/docs/formal/04-03-12.pdf (2004)
23. MicrosoftCorporation: Distributed Component Object Model Protocol-DCOM/1.0, draft, Microsoft Corporation. (1996)
24. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services: Concepts, Architectures and Applications, Springer. (2004)
25. Rysavy, P.: General Packet Radio Service (GPRS), GSM Data Today online journal. http://www.rysavy.com/Articles/GPRS/GPRS.htm (1998)
26. ETSI: GSM Technical Specification, GSM 03.64: General Packet Radio Service (GPRS). (1997)
27. Ericsson: Enhanced Data Rates for GSM Evolution (EDGE) - Introduction of high-speed data in GSM/GPRS networks, white paper, Ericsson AB. http://www.ericsson.com/technology/whitepapers/edge_wp_technical.pdf (2003)
28. Umtsworld: Overview of the Universal Mobile Telecommunication System, UMTS world. http://www.umtsworld.com/technology/overview.htm (2002)
29. 4G Press: World's First 2.5Gbps Packet Transmission in 4G Field Experiment, http://www.4g.co.uk/PR2006/2056.htm (2005)
30. SonyEricsson: Java support in SonyEricsson mobile phones P800 and P802, Developer guidelines from SonyEricsson Mobile CommunicationsAB, Jan. www.SonyEricssonMobile.com (2003)
31. kSOAP2: An open source SOAP implementation for kVM, http://ksoap.org/ (2007)
32. J2ME: Java 2 Micro Edition (J2ME), http://java.sun.com/javame/index.jsp (2007)
33. Hummel, J., Lechner, U.: Business models and system architectures of virtual communities. From a sociological phenomenon to peer-to-peer architectures, International Journal of Electronic Commerce, 6(3) (2002) 41–53
34. Srirama, S.: Publishing and Discovery of Mobile Web Services in Peer to Peer Networks, International Workshop on Mobile Services and Personalized Environments (MSPE'06), Aachen, GI. (2006) 99-112
35. Apte, Deutsch, Jain: Wireless SOAP: Optimizations for Mobile Wireless Web Services, http://www2005.org/cdrom/docs/p1178.pdf (2005)
36. Van Engelen, R., Galliva, K.: The gSOAP toolkit for web services and peer-to-peer computing networks, In 2nd IEEE International Symposium on Cluster Computing and the Grid. (2002)

37. eSOAP: eSOAP – Architecture. http://esoap.ultimodule.com/ (2007)
38. Sandoz, P., Pericas-Geertsen, S., Kawaguchi, K., Hadley, M., Pelegri-Llopart, E.: Fast Web Services, http://java.sun.com/developer/technicalArticles/WebServices/fastWS/ (2003)
39. ETSI: GSM Technical Specification, GSM 03.34, High Speed Circuit Switched Data (HSCSD). (1996)
40. Johnson, D., Perkins, C., Arkko, J.: Mobility Support in IPv6. IETF (2002)
41. Novak, L., Svensson, M.: MMS—Building on the Success of SMS, Ericsson Rev., No. 3. (2001) 102–109
42. GPS: Global Positioning System: Data format. U.S. Coast Guard Navigation Center, http://www.navcen.uscg.gov/pubs/gps/sigspec/gpssps1.pdf (2004)
43. Apache: Apache Axis, Apache Web Services Project, http://ws.apache.org/axis/ (2007)
44. 3GPP: 3GPP TS 24.011: Point-to-Point (PP) Short Message Service (SMS) support on mobile radio interface.
45. Chatti, M., Srirama, S., Kensche, D., Cao, Y.: Mobile Web Services for Collaborative Learning, in Proceedings of the 4th International Workshop on Wireless, Mobile and Ubiquitous Technologies in Education (WMUTE 2006). Athens, Greece (2006)
46. Belov, N., Braude, I., Krandick, W., Shaffer, J.: Wireless Internet Collaboration System on Smartphones, 3rd International Workshop on Ubiquitous Mobile Information and Collaboration Systems, UMICS 2005, a CAiSE'05 workshop (2005)
47. IETF: The SSL Protocol Version 3.0, Internet draft, IETF. http://www.freesoft.org/CIE/Topics/ssl-draft/INDEX.HTM (1996)
48. IETF: HTTP over TLS, IETF RFC 2818. http://www.ietf.org/rfc/rfc2818.txt (2000)
49. Meier, J.D., Mackman, A., Dunner, M., Vasireddy, S., Escamilla, R., Murukan, A.: Improving Web Application Security: Threats and Countermeasures, MSDN, Microsoft Corporation. (2003)
50. Reagle, J. et al: XML Encryption, W3C Working group note. http://www.w3.org/Encryption/2001/ (2001)
51. Eastlake, D., Reagle, J., Solo, D. et al.: XML-Signature Syntax and Processing, W3C Note. http://www.w3.org/TR/xmldsig-core/ (2002)
52. IBM: Security in a Web Services world: A Proposed Architecture and Roadmap, IBM Developerworks. (2002)
53. Lockhart, H., Parducci, B.: OASIS eXtensible Access Control Markup Language (XACML), OASIS Standard Specification. (2005)
54. LibertyAlliance: Liberty Alliance Project Whitepaper: Personal Identity, The Liberty Alliance. (2006)
55. OMA: OMA Web Services Enabler (OWSER): Overview, Open mobile alliance group. http://www.openmobilealliance.org/release_program/docs/OWSER/V1_0-20040715-A/OMA-OWSER-Overview-V1_0-20040715-A.pdf (2004)
56. JSR 118: Mobile Information Device Profile (MIDP) V2.0., Java Community process. http://java.sun.com/products/midp/ (2002)
57. JSR 139: Connected Limited Device Configuration (CLDC), Java Community process. http://java.sun.com/products/cldc/ (2002)
58. BouncyCastle: Bouncy Castle Crypto APIs, The Legion of the Bouncy Castle. http://www.bouncycastle.org/ (2007)
59. RSALabs: Cryptographic technologies, RSA Labs. http://www.rsasecurity.com/rsalabs/node.asp?id=2212 (2007)
60. TRIPLEDES: Triple Digital Encryption Standard, RSA Labs. http://www.rsasecurity.com/rsalabs/node.asp?id=2231 (2007)
61. AES: Advanced Encryption Standard, RSA Labs. http://www.rsasecurity.com/rsalabs/node.asp?id=2234 (2007)
62. Rivest, R., Shamir, A. and Adleman, L. M.: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, Communications of the ACM, v. 21, n. 2. (1978) 120-126

63. DSS: Digital Signature Standard, RSA Labs. http://www.rsasecurity.com/rsalabs/node.asp?id=2239 (2007)
64. Lai, X.: On the design and security of block ciphers, ETH Series in Information Processing, Massey, J.L. (editor), vol. 1, Hartung-Gorre Verlag Konstanz, Technische Hochschule (Zurich). (1992)
65. FIPS: Data Encryption Standard (DES), Federal Information Processing Standards Publication, October, FIPS PUB-43. http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf (1999)
66. Srirama, S., Jarke, M., Prinz, W.: A performance evaluation of mobile web services security, 3rd International Conference on Web Information Systems and Technologies (WEBIST 2007), Barcelona, Spain, INSTICC Press. (2007)
67. Laukkanen, M., Helin, H.: Web Services in wireless networks: What happened to the performance, in the proceedings of the Int. Conf. on Web Services – ICWS '03, CSREA Press. (2003) 278-284
68. Boyer, J.: Canonical XML, W3C Recommendation. http://www.w3.org/TR/xml-c14n, http://www.ietf.org/rfc/rfc3076.txt (2001)
69. Cokus, M., Pericas-Geertsen, S.: XML Binary Characterization Use Cases, W3C Working Group Note. http://www.w3.org/TR/xbc-use-cases/ (2005)
70. Chiu, K., Govindaraju, M., Bramley, R.: Investigating the Limits of SOAP Performance for Scientific Computing, 11th IEEE International Symposium on High Performance Distributed Computing HPDC-11 2002, IEEE Computing Society. (2002) 256
71. Schulte, R.: Predicts 2003: Enterprise service buses emerge, Report, Gartner. (2002)
72. Srirama, S.N., Jarke, M., Prinz, W.: A Mediation Framework for Mobile Web Service Provisioning, edocw, 10th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW'06). (2006) 14
73. Ripeanu, M., Foster, I., Iamnitchi, A.: Mapping the Gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. IEEE Internet Computing Journal, 6(1) (2002) 51–57
74. Anderson, D.P., Cobb, J., Korpela, E., Lebofsky, M., & Werthimer, D.: SETI@home: An experiment in public-resource computing. Communications of the ACM, 45(11) (2002) 56–61.
75. Carlsson, B., Gustavsson, R.: The Rise and Fall of Napster - An Evolutionary Approach. Proceedings of the 6th International Computer Science Conference on Active Media Technology. (2001) 347-354
76. Meta Search Inc: eDonkey2000 Home Page. http://www.edonkey2000.com/ (2007)
77. Pouwelse, J., Garbacki, P., Epema, D., Sips, H.: The bittorrent p2p file-sharing system: Measurements and analysis, in Proc. 4th International Workshop on Peer-to-Peer Systems (IPTPS'05), Ithaca, New York, USA. (2005)
78. Schneider, J.: Convergence of Peer and Web Services. http://www.openp2p.com/pub/a/p2p/2001/07/20/convergence.html (2001)
79. Bolcer, G.A., Gorlick, M., Hitomi, P., Kammer, A.S., Morrow, B., Oreizy, P., et al.: Peer-to-peer architectures and the Magi™ open-source infrastructure. from http://www.endeavors.com/pdfs/ETI%20P2P%20white%20paper.pdf (2000)
80. JXME: The JXTA Java Micro Edition. http://jxme.jxta.org/ (2007)
81. Cutting, D.: Apache Lucene. http://lucene.apache.org/ (2007)
82. JSR 172: J2ME Web Services Specification, Java community process. (2004)
83. W3C: Web Ontology Language (OWL). http://www.w3.org/TR/owlfeatures/ (2004)
84. W3C: Web Ontology Language for Services (OWL-S) 1.1. http://www.daml.org/services/owl-s/1.1/ (2007)