Mobile Hosts in Enterprise Service Integration

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Master of Science (M.Sc.)
Satish Narayana Srirama

aus Peddapadmapuram, A.P., India

Berichter: Universitätsprofessor Dr. Matthias Jarke, RWTH Aachen
Universitätsprofessor Dr. Wolfgang Nejdl, TU Hannover

Tag der mündlichen Prüfung: 19. September 2008

Abstract

Basic web services are widely being accessed from smart phones due to the advances in wireless devices and mobile communication technologies. While such mobile web service (MWS) clients are common, the thesis addresses the scope of providing web services from smart phones, thus integrating personalized mobile services into cellular enterprises and Internet. A 'Mobile Host' capable of providing basic web services from smart phones was developed and the performance analysis of the tool proved the technical feasibility of the concept.

Service delivery and management from Mobile Host are now technically feasible. However, the ability to provide proper quality of service (QoS) for the Mobile Host is observed to be very critical, especially in terms of security and reasonable scalability. In the security analysis, the adaptability of the WS- Security specification to the mobile web service provisioning (MWSP) domain is analyzed. The scalability analysis tried to reduce the size of the MWS messages being transmitted by means of different compression technologies. The analyses mainly focused on additional latencies caused to the performance of the Mobile Host by the added QoS features.

Mobile Host finds its use in domains like mobile community support, collaborative learning, social systems etc. The smart phone can act as a multi-user device without additional manual effort on part of the mobile user. The thesis extensively studied the applications of Mobile Host in the m-learning domain and developed the MobileHost CoLearn system, as a tool helping in the collaborative learning process. The user evaluation of the system clearly showed the adaptability of Mobile Hosts by different communities.

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. The traditional centralized UDDI (Universal Description, Discovery and Integration) based registries are not adaptable, as they can have bottlenecks and can make single points of failure. Moreover, the dynamic nature of the mobile nodes makes the binding information change regularly, thus leaving many stale advertisements in the registry. Hence a dynamic MWS discovery mechanism is proposed, that uses the peer to peer network and its features for publishing and discovery of mobile web services.

Based on Mobile Host's application, QoS and discovery research, the deployment scenario of the Mobile Hosts in cellular networks is identified. A 'Mobile Web Services Mediation Framework (MWSMF)' is established as an intermediary between the web service clients and the Mobile Hosts, based on the Enterprise Service Bus (ESB) technology.

Kurzfassung

Durch neuere Entwicklungen im Bereich mobiler Geräte haben einfache Web Services, die von Smart Phones aus verwendet werden können, weite Verbreitung gefunden. Während aber MWS Clients bereits weit verbreitet sind, befasst sich diese Arbeit mit der Problematik, Web Services auf Smart Phones bereitzustellen, also mit der Integration personalisierter mobiler Dienste in Mobilfunknetze und das Internet. Es wurde ein 'Mobile Host' entwickelt, der es ermöglicht, einfache Web Services durch Smart Phones anzubieten. Die Performanz-Analyse des Systems hat die technische Machbarkeit des Konzepts bewiesen.

Bereitstellung und Verwaltung von Diensten auf mobilen Geräten sind nun technisch machbar. Es hat sich allerdings gezeigt, dass es für den Mobile Host von besonderer Bedeutung ist, eine angemessene Dienst-Qualität (QoS, "quality of service") zu garantieren, insbesondere in Bezug auf Sicherheit und Skalierbarkeit. Bei der Sicherheitsanalyse wurde die Anpassbarkeit der WS-Security-Spezifikation an den Bereich der Bereitstellung mobiler Web Services (mobile web service provisioning, MWSP) analysiert. In der Skalierbarkeitsanalyse wurde mit Hilfe verschiedener Komprimierungstechniken versucht, die Größe der übermittelten MWS-Nachrichten zu reduzieren. Der Fokus der Analyse wurde auf die zusätzlichen Latenzzeiten im Mobile Host gelegt, die durch die QoS Merkmale verursacht werden.

Anwendungen für Mobile Hosts lassen sich unter anderem in den Bereichen Mobile Community Support, Collaborative Learning und Social Systems finden. Das Smart Phone kann ohne zusätzlichen manuellen Aufwand durch den Anwender als Mehr-Benutzer-Gerät agieren. Im Rahmen der Arbeit wurden Anwendungen für Mobile Hosts im Bereich des m-learning untersucht. So wurde das MobileHost CoLearn System als Werkzeug zur Unterstützung des kollaborativen Lernprozesses entwickelt. Die Nutzer-Evaluaierung des Systems hat die Anpassbarkeit von Mobile Hosts durch verschiedene Communities klar gezeigt.

Die hohe Zahl möglicher Web Services, wobei jeder Mobile Host mehrere Dienste im Netzwerk anbieten kann, macht die Ermittlung (discovery) dieser Dienste sehr komplex. Traditionelle zentralisierte UDDI-basierte (Universal Description, Discovery and Integration) registries sind nicht anpassbar, da sie Flaschenhälse aufweisen können und als 'single point of failure' anfällig sind. Des Weiteren erfordert die dynamische Natur von mobilen Knoten regelmäßige Änderungen der Information über Dienste, wodurch viele veraltete Advertisements in der Registrierung zu finden sind. Daher wird alternativ ein dynamischer MWS-discovery-Mechanismus vorgestellt, der das Peer-to-Peer Netzwerk und seine Merkmale für Publishing und Discovery von Mobile Web Services nutzt.

Auf Grundlage bestehender Forschung zu Anwendungen, QoS und Discovery von Mobile Hosts wird ein Deployment-Szenario für Mobile Hosts in Mobilfunknetzen identifiziert. Basierend auf der Enterprise-Service-Bus-Technologie (ESB) wird ein 'Mobile Web Services Mediation Framework (MWSMF)' als Schnittstelle zwischen den Web Service Clients und den Mobile Hosts eingeführt.

Acknowledgements

Ever since my engineering days, I always wanted to pursue higher education and head for Doctorate degree. I am fortunate that I could achieve this goal at such a prestigious institute as Rheinisch-Westfaelische Technische Hochschule Aachen (RWTH Aachen). There are several people without whom this dream would not have been a reality. I would like to express my deep and sincere gratitude to all those involved.

First of all, I thank Prof. Matthias Jarke for all the support he has extended me, during the past six years. I would also like to thank him for all the confidence he had shown, in me and my work. Secondly, I thank Prof. Wolfgang Nejdl for reviewing the thesis and being my second examiner. Nonetheless I would also like to thank Prof. Wolfgang Prinz and owe him lots of gratitude for having shown me this way of research.

I would also like to thank Friedhelm Ramme, Roman Levenshteyn and Martin Gerdes of Ericsson Research for their help and support, during the research. I am especially thankful to Roman Levenshteyn for all his advice, positive criticism, and willingness to discuss any questions or ideas that I have had, during the past years.

I would also like to thank all my co-authors: Prof. Matthias Jarke, Prof. Wolfgang Prinz, Mohamed Amine Chatti, David Kensche, Yiwei Cao, Dr. Ralf Klamma, Anton Naumenko, Pravin Pawar, Bert-Jan van Beijnum, Aart van Halteren, Iliyana Ivanova, Hongyan Zhu, Kiran Kumar Pendyala, Kaifei Wang, Rajsekhar Kakumani, Akshaia Aggarwal and Prof. Vagan Terziyan. I am thankful to all my students, whose work contributed my research.

The work was initially sponsored by Graduiertenkollegs - "Software für mobile Kommunikationssysteme" at RWTH Aachen, later by Ultra High-Speed Mobile Information and Communication (UMIC) excellence research cluster, established under the excellence initiative of the German government, and partly supported by Ericsson research. I thank all the funding agencies and appreciate their financial support for my projects.

I am also grateful to the Information Systems & Databases Group, at RWTH Aachen for providing me an excellent work environment during the past years. I thank all my colleagues, a slight banter with whom, always made me better focused.

I sincerely thank my parents for their never ending support and especially my father, Prof. Lakshminarayana Srirama, for being an inspiration and role model. The blessings of my mother Lolakshi Srirama and the encouragement of my brother Jagadeesh Narayana Srirama, sister Ambika Meesala and brother-in-law Ramesh Naidu Meesala are sincerely acknowledged. Special thanks are to my beloved wife, Gayatri, for her compassion and patience during the last days of my dissertation.

Finally I like to thank my friends and relatives for having them on my side, especially Pravin Pawar and Prof. Lakshmana Das Nadukuru for our fruitful discussions before considering the doctoral studies.

Aachen
22nd September, 2008

Satish Narayana Srirama

Contents

List of Figures

List of Tables

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 led 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(eXtensible Markup Language)-based open standards and transport protocols such as SOAP (Simple Object Access Protocol) and HTTP (HyperText Transfer Protocol). Public interfaces of web services are defined and described using W3C (World Wide Web Consortium) based standard, Web Service Description Language (WSDL). 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 [Gottschalk et al., 2002].

Concurrently, the capabilities of high-end mobile phones and PDAs (Personal Digital Assistant) 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 higher data transmission rates achieved in wireless domains with Third Generation (3G) and Fourth Generation (4G) 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.

1.1 Motivation

To meet the 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 several organisations such as Open Mobile Alliance (OMA), Liberty Alliance (LA) 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 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.

Mobile terminals accessing the web services cater for anytime and anywhere access to services. Some interesting mobile web service applications are the provisioning of services like e-mail, information search, language translation, company news etc. for employees who travel regularly. There are also many public web services accessible from smart phones like the weather forecast, stock quotes etc. Mobile web service clients are also significant in the geospatial and location based services [Benatallah and Maamar, 2003]. While *mobile web service clients* are common, the combination of the cellular and web services domains would only be completed, if it is feasible to also offer standard *web service providers* on smart phones.

The paradigm shift of smart phones from the role of service consumer to the service provider is a step towards practical realization of various computing paradigms such as pervasive computing, ubiquitous computing, ambient computing and context-aware computing. For example, the applications hosted on a mobile device provide information about the associated user (e.g. location, agenda) as well as the surrounding environment (e.g. signal strength, bandwidth). Mobile devices also support multiple integrated devices (e.g. camera) and auxiliary devices (e.g. Global Positioning Systems (GPS) receivers, printers). For the hosted services, they provide a gateway to make available their functionality to the outside world (e.g. providing paramedics assistance). In the absence of such provisioning functionality the mobile user has to regularly update the contents to a standard server, with each update of the device's state. The scope of *mobile web service provisioning* was studied by two projects at RWTH Aachen University since 2003 [Srirama et al., 2006d; Gehlen, 2007], where *Mobile Hosts* were developed, capable of providing basic web services from smart phones. Once the Mobile Host was developed, an extensive performance analysis was conducted to prove its technical feasibility [Srirama, 2004].

While service delivery and management from Mobile Host were thus shown technically feasible, the ability to provide proper Quality of Service (QoS), especially in terms of security and reasonable 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. In terms of scalability, the Mobile Host has to process reasonable number of clients, over long durations, without failure and without seriously impeding normal functioning of the smart phone for the user. Similarly 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. This thesis addresses these issues.

1.2 Contributions

For the traditional wired networks and web services, a lot of standardized security specifications, protocols and implementations like WS-Security, Security Assertion Markup Language (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. The thesis 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. The study analyzed the adaptability of WS-Security to the mobile web service provisioning domain. Mainly it observed the additional 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.

In terms of scalability for the Mobile Host, the layered model of web service (WS) communication, introduces message overhead to the verbose XML based SOAP messages. This consumes lot of resources, since all this additional information is to be exchanged over the radio link. Many compression techniques are 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 compression of the SOAP messages needs some extra processing power at the smart phones and thus adds further performance latencies. The performance penalties of different compression techniques are studied in detail. The study has observed BinXML as a good compression option and the binary encoding was adapted for the mobile web service invocation cycle.

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. Several prototypical 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 (Really Simple Syndication) feeds provided by a server. The thesis has also studied the applications of Mobile Host in the *m-learning* domain and with the user evaluation of the MobileHost CoLearn system, the study clearly showed the adaptability of Mobile Hosts by different communities.

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. The traditional centralized UDDI (Universal Description, Discovery and Integration) based registries 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, the thesis proposed an alternative approach of discovering mobile web services. The *mobile web services discovery* mechanism uses the *peer to peer (P2P)* [Milojicic et al., 2003] network for advertising the mobile web services and depends on the P2P network for discovering the services. A prototypical was developed using the JXTA (Juxtapose) network [Wilson, 2002], and it was able to publish mobile web services and discover them from the smart phones, with reasonable performance latencies.

Based on the Mobile Host's application, QoS and discovery research, the thesis has identified the deployment scenario of the Mobile Hosts in the cellular networks. A *Mobile Web Services Mediation Framework (MWSMF)* is established as an intermediary between the web service clients and the Mobile Hosts in the JXTA network, based on the *enterprise service bus (ESB)* technology. The features, realization details and benefits of the mediation framework are studied in detail with different scenarios. The MWSMF is also extensively analyzed for its scalability, so that it can handle large number of concurrent clients, possible in mobile operator networks.

1.3 Outline

Chapter 2 discusses the state of the art for the research addressed by the thesis. The chapter first introduces the web services technology along with associated standards and protocols. Later it introduces the wireless developments in terms of device and transmission capabilities thus paving the way for an interesting discussion on nomadic mobile services and mobile web services. Here the supported devices and platforms, standardization efforts and implementations, and SOAP transmission mechanisms for mobile web services are discussed in detail. The chapter concludes with a discussion of the current generation telecommunication technologies.

Chapter 3 explains the research with mobile web service provisioning and tries to provide motivation and state of the art for the upcoming chapters. First it introduces the mobile web service provisioning concept and the developed Mobile Host with its performance and application analysis. Similar means of providing services from smart phones using alternate technologies are also discussed in detail.

The chapter later discusses the security and scalability related standards and specifications for mobile web services. Then it introduces the convergence of web services and P2P technologies, and the related projects addressing this issue, along with the basics of P2P technology and JXTA platform. The chapter finally discusses the integration issues for mobile web services and introduces the ESB technology, the Java Business Integration (JBI) specification and the open source ServiceMix tool used in realizing the integration framework.

Chapter 4 discusses the QoS extensions for the Mobile Host. The chapter first introduces the security analysis of the Mobile Host addressed by the thesis. Here the security analysis's design, implementation and evaluation models are explained in detail

considering the security challenges associated with mobile web services. The chapter later discusses the evaluation results and suggests the best means of securing mobile web services being provided from smart phones. Further discussions of the end point security of mobile web services lead to Semantic-Based Access Control (SBAC) mechanism. The SBAC mechanism and its deployment options are explained in detail, with the developed prototype.

After the security analysis discussion the chapter introduces the scalability analysis of the Mobile Host addressed by the thesis. Here the design, implementation and evaluation models of Mobile Host's scalability are explained in detail along with the evaluation results.

Chapter 5 discusses the details of Mobile Host's entry into P2P networks. First, the concept of mobile web service provisioning in P2P networks is explained in detail. The chapter later introduces the technical advantages achieved with P2P, for the mobile web service provisioning, in terms of better mobile terminal access and mobile web service discovery mechanisms. The chapter especially concentrates at the discovery mechanisms and discusses the mobile web service discovery mechanism, proposed by the thesis. Here the discovery approach, its categorization and context awareness issues and evaluation results are addressed in detail. The chapter concludes with an overall discussion of publishing, discovery and invocation of mobile web services in JXME (JXTA for J2ME (Java 2 Platform, Micro Edition)) networks.

Chapter 6 discusses the details of integration framework required for mobile web service provisioning in commercial mobile operator proprietary networks. The chapter first proposes the deployment scenario and later produces the realization details of the MWSMF. Here the components and message flows of different scenarios of the MWSMF are explained in detail, along with the supplementary features it provides for the mobile web service provisioning. The chapter concludes with a detailed evaluation of the MWSMF, proving that the mediation framework can scale to handling large number of concurrent clients, possible in mobile operator networks.

Chapter 7 discusses the application domains of Mobile Host and introduces some of the applications developed by the thesis. The chapter mainly focuses at the m-learning domain and describes the realized MobileHost CoLearn system in detail. Later sections of the chapter provide the detailed performance and usability evaluation of the system, proving the adaptability of Mobile Hosts by different communities. The user evaluation questionnaire used in this analysis is provided in Appendix B.

Chapter 8 concludes the findings of the thesis and discusses the future research directions associated with this research. The chapter also provides the experience of the study with using open source tools and softwares.

2 State of the Art

It is well accepted by now that the Internet can be seen as a large-scale distributed information system with numerous information users and providers. From the view-point of information systems engineering, this has influenced three major trends:

- the evolution from static content to web services,
- the evolution from client-server systems to peer-to-peer and pervasive computing systems,

and finally - in conjunction with novel developments in wireless communication technology -

- the trend from stationary to mobile distributed information management.

This chapter introduces these developments from the literature and addresses the motivation for the research in mobile web services domain.

2.1 Web Services

Service Oriented Architecture (SOA) [Burbeck, 2000; Endrei et al., 2004] is a trend in information systems engineering and the software industry's response to the problem of managing large monolithic applications. SOA is a component model that delivers application functionality as services to end-user applications and other services, bringing the benefits of loose coupling and encapsulation to the enterprise application integration. Services encapsulate reusable business function and are defined by explicit, implementation-independent interfaces. Services are invoked through communication protocols that stress location transparency and interoperability. SOA defines participating roles as, *service provider*, *service client*, and *service registry*. Figure 2.1 on the next page shows the SOA collaborations. The operations publish, find, bind and the artifacts *services* and *service descriptions* are all shown in the figure. SOA is not a new notion and many technologies like CORBA (Common Object Request Broker Architecture) [OMG, 2004] and DCOM (Distributed Component Object Model) [Brown and Kindel, 1998] at least partly represent this idea. Using web services for SOA provides certain advantages over other technologies. Specifically, web services are based on a set of still evolving, though well-defined W3C standards, that allow much more than, just defining interfaces.

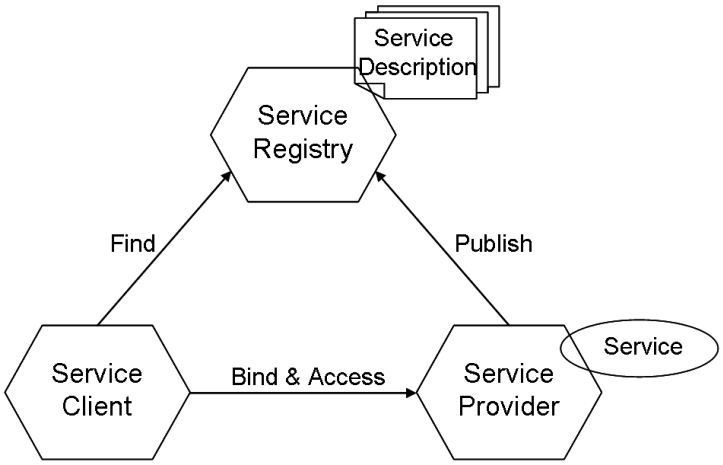


Figure 2.1: Service oriented architecture collaborations (Redrawn from [Gottschalk et al., 2002])

Web services are self-contained modular applications, which can be described, published, located and invoked over a network, generally, the World Wide Web (WWW). They are changing the Internet from program-to-user business-to-consumer interactions to program-to-program business-to-business interactions. Web services technology and its protocol stack are based on open standards and are widely accepted in 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 service technology 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. [Alonso et al., 2004]

2.1.1 Web Services Architecture

Web services architecture is defined by the Web Services Activity (WSA) [Booth et al., 2004] and enables application-to-application communication over the Internet. The goal of the Web Services Activity is to develop a set of technologies in order to lead web services to their full potential [Gottschalk et al., 2002]. Public interfaces of web services are defined and described using Web Services Description Language (WSDL) and web services allow access to software components through standard Web technologies and protocols like SOAP and HTTP [Fielding et al., 1999], regardless of their platforms, implementation details. By following the same architecture as SOA, as shown in figure 2.1, a service provider develops and deploys the web service and publishes its description and binding/access details (WSDL) with a public registry, generally a UDDI based registry. Any potential client queries the UDDI registry (Find) and retrieves the service description.

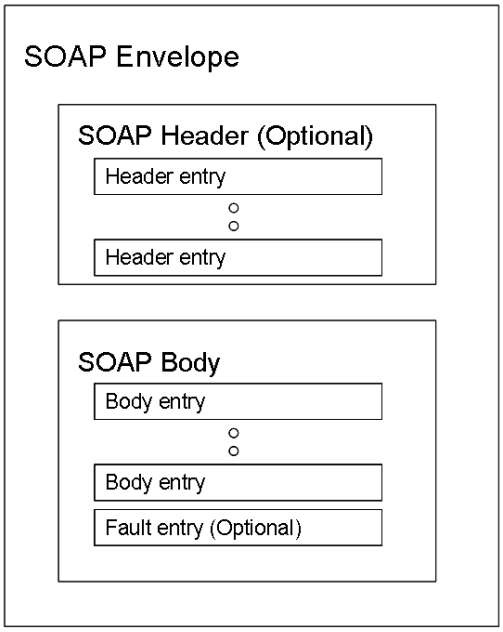


Figure 2.2: SOAP message structure

After the WSDL has been retrieved, the service requester binds to the service provider by invoking the service through SOAP. The communication between client and UDDI registry is also based on SOAP [Curbera et al., 2002]. The following subsections introduce web services protocols, standards and their industrial adaptability efforts of Web Services Interoperability (WS-I) organization.

2.1.2 Simple Object Access Protocol (SOAP)

SOAP is a lightweight XML based protocol for exchanging structured information between peers in a decentralized and distributed environment [Gudgin et al., 2007]. It provides messages to communicate between applications running on different operating systems, with different technologies and programming languages. A SOAP message is an XML document that consists of a mandatory SOAP Envelope, which contains an optional *SOAP Header* and a mandatory *SOAP Body*. The SOAP message structure is shown in figure 2.2.

Envelope is the root element of the SOAP message. It specifies two things: an XML namespace and an encoding style. The *namespace* declaration gives a clue for the used SOAP version. SOAP versions 1.1 and 1.2 are almost the same, and the complete set of differences between them can be viewed at [Mitra and Lafon, 2007]. The *encodingStyle* attribute indicates the serialization rules, used in the message, which can be explicitly overridden in the child elements of Envelope.

Header is an optional sub-element of Envelope, which is a flexible mechanism for

extending the SOAP message in a decentralized manner, without prior agreement between the communicating parties. The header entries contain information, such as authentication information, digital signatures, transaction management details, payment details etc. There are a couple of defined attributes for the header entries, such as *actor*, which indicates the recipient of the header element, and *mustUnderstand*, which indicates whether successful processing of the header entry by the recipient is mandatory.

Body is a mandatory sub-element of Envelope, which encompasses the message payload intended for the recipient of the SOAP message. Generally, it contains marshaled RPC (Remote Procedure Call) calls or error reports. In the case of a request message the payload of the message is processed by the receiver of the message and is typically a request to perform some service and, optionally, to return some results. In the case of a response message the payload is typically the results of the request or a fault. The optional **Fault** sub-element of SOAP body element specifies error information. An error could be generated at any SOAP intermediary along the message path while processing SOAP message.

2.1.3 Web Services Description Language (WSDL)

Web Services Description Language (WSDL) is an XML based specification that defines the means of describing web services [Christensen et al., 2001]. It specifies the Interface information describing all available public functions, data type information for all message requests and message responses, binding information about the transport protocol to be used, address information for locating the specified web service, and etc. Using WSDL, a client can locate a web service and invoke any of its publicly available functions. The process can also be automated, enabling applications to easily integrate with new services with little or no manual code and interaction. A WSDL document describes a web service as a collection of abstract items called *ports* or *endpoints*. The WSDL specification uses the following main elements in the definition of network services: *definitions*, *types*, *message*, *portType*, *binding* and *service*. The WSDL 2.0 version uses *interface* and *endpoint* terms, instead of portType and port respectively used by WSDL 1.1. The structure of WSDL is shown in figure 2.3 on the facing page.

Definitions is the root element of every WSDL document. It defines the name of the web service, declares multiple namespaces used throughout the remainder of the document, and contains all the sub-elements, described below.

Types is a sub-element of Definitions and the container of all data type definitions. The element describes all the data types exchanged between the web service consumer and the web service provider. The complex data types and the corresponding user defined data types are represented using some type system, such as XSD (XML Schema Definition). If the service uses only XML schema built-in simple types, such as strings and integers, the Types element is not required.

Message is an abstract typed definition of the data being communicated. It defines all the input and output parameters of the publicly available functions of a service. The

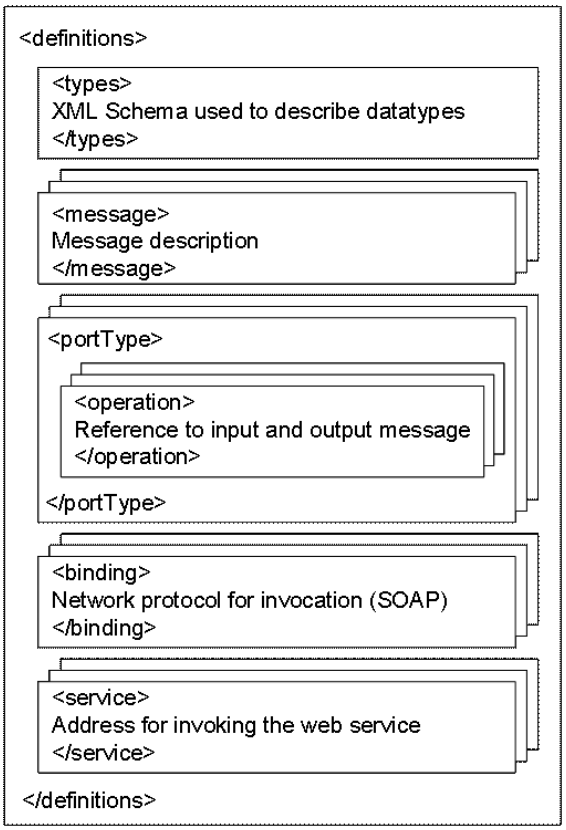


Figure 2.3: The structure of WSDL document

element specifies the name of the message and contains zero or more message part elements.

PortType is an abstract set of operations, supported by one or more ports. It is the most important element of the WSDL document, as it defines the web service. The element specifies the operations that can be performed, and the messages that are involved for the service. **Operation** is a sub-element of PortType and gives an abstract description of an action supported by the service.

Binding element describes the concrete specifics of how the service will be implemented on the wire. WSDL includes built-in extensions for defining SOAP services, and hence the SOAP-specific information is incorporated in this element.

Service is a collection of related ports. It defines the address for invoking the specified web service. **Port** is a sub-element of Service, which is a single endpoint, defined as a combination of a binding and a network address.

2.1.4 Universal Description, Discovery and Integration (UDDI)

Universal Description, Discovery and Integration (UDDI) is a cross-industry effort, driven by all major platform and software providers, such as Dell, Fujitsu, HP, IBM, Intel, Microsoft, SAP, Oracle, Sun and Hitachi. UDDI is the name of a group of web-based registries, which expose information about an entity, be it a business entity or another entity, and its technical interfaces, or APIs (Application Programming Interface). These registries are run by multiple operator sites, whose basic services can be accessed by anyone, free of charge [Bellwood, 2002].

From a business developer's point of view, UDDI is similar to an Internet search engine, which can be used to browse UDDI registries to view different businesses that expose web services, and the specifications of those services. Software developers can use the UDDI Programmers API (Application Programming Interface) to query the registry to discover services matching different criteria. Both business developers and software developers can publish new business entities and services to the UDDI registry. Conceptually, a business can register three types of information into a UDDI registry.

- *White pages* with basic contact information and identifiers about a company, including business name, address, contact information and unique identifiers. This information allows clients to discover web services based upon the business identification.
- *Yellow pages* with information that describes a web services using different categorizations (taxonomies).
- *Green pages* with technical information that describes the behaviors and supported functions of a web services, hosted by your business. This information includes pointers to the grouping information of web services and where the web services are located.

The information that makes up a registration consists of following data structure types. These data structures, passed as input and output parameters of major API messages, to and from the UDDI registries, are shown in figure 2.4 on the next page, and are briefly described here.

Business Entity represents basic information about a business, including contact data, categorization, identifiers, descriptions, etc. Publisher Assertion is used to establish public relationships between two Business Entity structures. Such a relationship is visible to the public, only if both companies have created the same Publisher Assertion documents separately, i.e. business relationships can not be one-sided.

Business Service represents a single, logical service classification, including information about how to bind to a service, what type of service it is, etc. A Business Entity can contain one or more Business Service structures.

Binding Template contains the technical descriptions of the web services, represented by the Business Service structure. It also contains the access point URL (Uniform

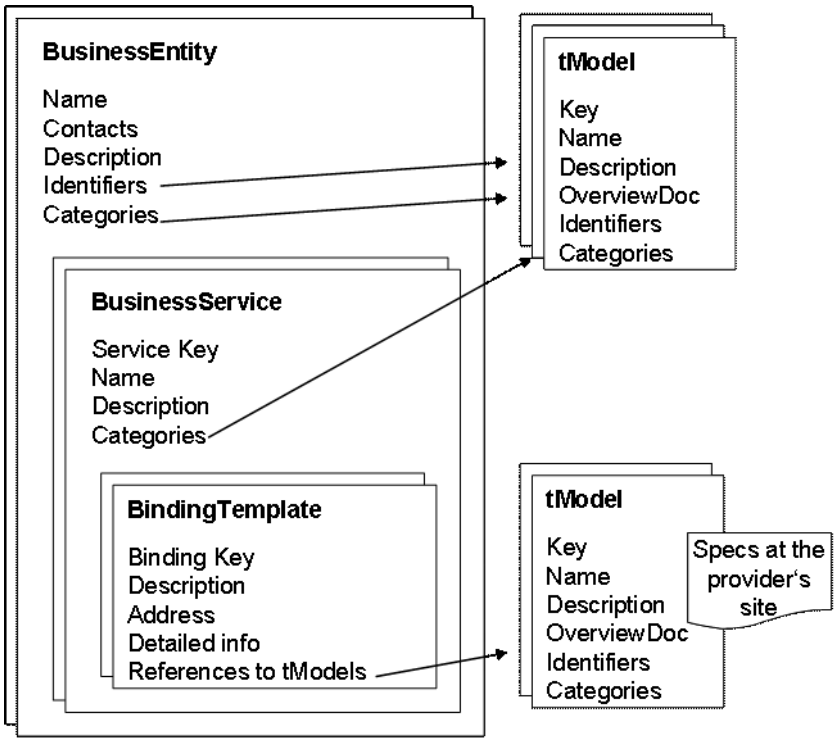


Figure 2.4: A schematic view of a UDDI registry entry

Resource Locator) of the web service, but does not contain the service specification details. A Business Service can contain one or more Binding Templates. It is similar to the <service> element of the WSDL, described above.

TModel is an abstract description of a particular specification or behavior, to which the web service adheres. For example, a TModel can be defined to represent a portType, defined by the WSDL. Then a business service, implementing the portType, can be specified by associating the TModel with one of the binding templates of the business service.

2.1.5 Web Services Interoperability

In order to facilitate the development of truly interoperable web services, the Web Services Interoperability organization (WS-I) [WS-I, 2004] was formed in February 2002. WS-I is an open, industry consortium of about 150 companies of diverse industries like automotive, consumer packaged goods, finance, government, insurance, media, telecommunications, travel and the computer industry. WS-I, as a standards integrator, supports the relationships with standards bodies who own specifications and fosters communication and cooperation with industry consortia and other organizations. The main goal of the consortium is to encourage web service adoption and accelerating interoperable web service development

by providing guidance, best practices and other resources.

WS-I delivers a collection of profiles, which are descriptions of conventions and practices for the use of specific combination of web services through which systems can interact, that support technical requirements and specifications to achieve interoperable web services. The consortium is also responsible for providing use cases, usage scenarios, sample applications and testing tools. WS-I has finalized the *Basic Profile*, *Attachments Profile* and *Simple SOAP Binding Profile*. WS-I is also working on a *Basic Security Profile* for web services. WS-I Basic Profile Version 1.2 [Ballinger et al., 2007] supports the specifications like SOAP 1.1, WSDL 1.1, UDDI 2.0, WS-Addressing etc. The profile backs three usage scenarios, where a usage scenario is a design pattern of interacting entities like actors, roles and message exchange patterns [Werden et al., 2003]. They are 1.) *One-way* usage scenario, which is to be used when loss of information can be tolerated. 2.) *Synchronous request/response* usage scenario. 3.) *Basic callback* usage scenario simulating an asynchronous operation using synchronous operations. The Attachment profile complements the Basic Profile 1.1 to add support for conveying interoperable, SOAP with Attachments (SwA) with SOAP messages. Simple SOAP Binding Profile is derived from Basic Profile requirements related to serialization of an envelope and its representation in the message.

2.2 Developments in the Wireless Domain

Concurrent to the SOA developments, the capabilities of today's high-end mobile phones and PDAs have increased significantly, both in terms of processing powers and memory capabilities. 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 main driving force for the rapid acceptance of such small mobile devices is the capability to get services and run applications at any time and at any place, especially while on the move [Helal et al., 1999]. The experience from Japanese market shows that the most important factor in this development is that the terminals are permanently carried around, and thus people can use so-called "*niche-time*" to use the devices for various things [Ichikawa, 2002]. The telecom industry estimated that there are around 3 billion mobile users by winter 2007. According to some market analysis nearly half of the mobile devices available were internet-enabled in 2006 and the tendency is still growing [ETC, 2007]. The market capture of such smart phones is quite evident and in 2003 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 (Personal Computers) sold [Rollman and Schneider, 2004]. Similarly, the analysis from Gartner, Inc. states that in combination 42.1 million smart phones and PDAs were sold during first half of 2006. In 2005, smart phones outsold PDAs by a factor of 3.4 to 1. According to [CIA, 2006], smart phone sales will continue to grow and will surpass PDA sales by an 11:1 margin in 2011. This means that at least 500 million, perhaps nearly one

billion Internet-enabled smart phones will be in use in the world by the end of 2008.

2.2.1 Device Capabilities

Traditionally, the hand-held devices have many resource limitations like low computation capacities, limited storage capacities, and small display screens that could only display few lines of text with poor rendering quality. The new smart phones have larger graphics-oriented screens with support for colors, which enhance the wireless experience and entice the cellular users to exploit them for different services. They are also being provided with built-in cameras, infrared ports that can be used to control home devices and even fingerprint sensors for secure transactions [NTT DoCoMo, 2003].

From the hardware aspects of smart phones, most of these mobile devices are using CPUs (Central Processing Unit) which are based on *ARM (Advanced RISC Machine) architecture*. The ARM architecture is a 16/32-bit RISC (Reduced Instruction Set Computing) processor architecture which is being widely used in embedded designs. ARM processors offer combination of advanced logic, robust functionality and energy efficiency, at low cost and simpler designs enabling easy integration. Smart phones are specifically using *ARM9* series processors. ARM9 [ARM, 2005] processor can deliver up to 220MIPS (Million instructions per second) at 200MHz (Megahertz) on a 0.18μm process. These high speeds enable mobile devices to swiftly operate more complex data and thus tremendously increase their computability. All ARM9 family processors feature the *Thumb* compressed instruction set and EmbeddedICE JTAG-based (Joint Test Action Group) software debug logic. Apart from smart phones ARM9 processors are used in automotive control, instrumentation, safety systems, set-top boxes, high-end printers, PDAs and multimedia formats such as MP3 (MPEG-1 Audio Layer 3) audio and MPEG4 (Moving Picture Experts Group) video.

There are also breakthroughs in the memory capabilities of smart phones, like the NAND flash memory. Traditionally SDRAM (Synchronous Dynamic Random Access Memory) and NOR-based flash memory are used in smart phones. The transition to NAND flash memory has huge advantage in performance. NAND is 60 times faster and 80 times less energy hungry than NOR [Greenberg, 2005]. All these improvements expand the utility of mobile devices and fulfill user's demand (in terms of device performance) and thus expanding their user base.

2.2.2 Transmission Capabilities

Concurrent to the device capabilities, the data transmission rates across the wireless network also have increased significantly. Traditionally the second-generation (2G) GSM (Global System for Mobile communications) networks delivered high quality and secure mobile voice and data services like SMS (Short Message Service), circuit switched Internet access etc., with full roaming capabilities and across the world. The GSM

platform is a widely successful wireless technology. But, with the advent of the interim-generation (2.5G) technologies like GPRS (General Packet Radio Service) [Rysavy, 1998; ETSI, 1997] and EDGE (Enhanced Data rates for GSM Evolution) [Ericsson, 2003a], and third-generation (3G) technologies [3GPP, 2007] like UMTS (Universal Mobile Telecommunications System) [Umtsworld, 2002], still higher data transmission rates are achieved in the wireless domain, in the order of few hundreds of Kbs to 2 Mbs. The advent of Fourth Generation Wireless Services (4G) technologies and their deployment in south Asian countries suggests that mobile data transmissions of the rate of few GB (Gigabyte) is also possible [4GPress, 2005]. More details about these telecommunication technologies are addressed in section 2.5 on page 31. These higher data transmission rates achieved in wireless domains with 3G and 4G technologies along with the fast creeping of all-ip broadband based mobile networks further boosted the growth in the cellular market.

2.2.3 Nomadic Mobile Services

The developments in device capabilities and data transmission rates brought out a large scope and demand for software applications for smart phones in high-end wireless networks. Many software markets have evolved like NTT DoCoMo [NTT DoCoMo, 2007d] capturing this demand of this large mobile user base. Many nomadic services were provided to the mobile phone users. For Example, DoCoMo provides phone, video phone, *i-mode* (internet), and mail (i-mode mail, Short Mail, and SMS) services. i-mode is NTT DoCoMo's proprietary mobile internet platform [NTT DoCoMo, 2007c]. With i-mode, mobile phone users can get easy access to thousands of Internet sites, as well as specialized services such as e-mail, online shopping, mobile banking, ticket reservations, and restaurant reviews. Similarly, a free mapping, search and navigation application for mobile phones is being provided by LocatioNet Systems [LocatioNet, 2007]. The company's free service called *Amaze* looks like a hybrid between the popular *TomTom GPS (Global Positioning System) system* [TomTom, 2007] and *Google Maps* [Google, 2007].

Similarly, Google provides its local search tool, wireless search service [Google Mobile, 2007], designed specifically for users of mobile devices, in particular travelers. Travel tools like the *BlackBerry*, a wireless email device, provides the ability to be permanently online, with instant access to email. PayPal provides a mobile payment service, that lets mobile users send and receive funds on their cell phone via text message. Apart from these services many location based services (LBS) have been developed in improving the general tourism experience. Further with the start of initial 4G services to consumers, 4G networks can offer a range of new choices in the world of wireless services, including even higher speed mobile Internet access, video and TV in an on-the-go, real time, on-demand format and new options for networking and entertainment for the home. Ultimately, 4G will allow consumers to select a single provider to deliver their entire home, mobile and entertainment services [ETC, 2007].

These nomadic services bring benefits to all the participants of the mobile web. The

mobile users benefit from these mobile services and the mobile phone becomes the network computer and wallet PC (Personal Computer) for him. The *enterprises* can benefit as they can support technologies and services that allow for anywhere and anytime connectivity of the office information sources. The *mobile operator networks* can increase their revenues with "open" models. For example NTT DoCoMo with its i-mode portal has proved this success, where the operator provides a framework and environment in which third party content developers can deploy their services [NTT DoCoMo, 2007b]. The *content providers* can in turn get incentives from these open models. But with the breadth of information and service types that wireless users wish to access, discussed from previous paragraph, no mobile operator can provide all the required solutions.

From the analysis of most of these nomadic mobile services; each operator provided some set of services, applicable to specific group, over specific platforms. But most of these approaches were proprietary and followed specific protocols. For example if we consider a company trying to advertise itself, it can use the mobile push services that are run over the GSM network. Then the advertisement has to be shaped in such a way that it fits the terminals and platforms by the mobile operators and vendors. This makes the services un-interoperable and the integration of services becomes highly impossible. So in order to overcome the interoperability issues 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.

2.3 Mobile Web Services

In the 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. By following the basic web services architecture, 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 several organisations such as OMA [OMA, 2004, 2006a], LA [Tourzan and Koga, 2006] on the specifications front; some practical data service applications such as over-the-air provisioning (OTA), application handover etc. on the commercial front; and SUN, IBM toolkits [Sun Microsystems, 2007f], [IBM Corporation, 2007a] on the development front. Thus, though this is early stages, we can safely assume that mobile web services are the road ahead. Figure 2.5 on the next page shows the deployment scenario of mobile web services, where mobile devices are used as both web service providers and clients. Before considering the *mobile web service provisioning* in detail, this section introduces some of the mobile web services, and the platforms and APIs supporting the mobile web service

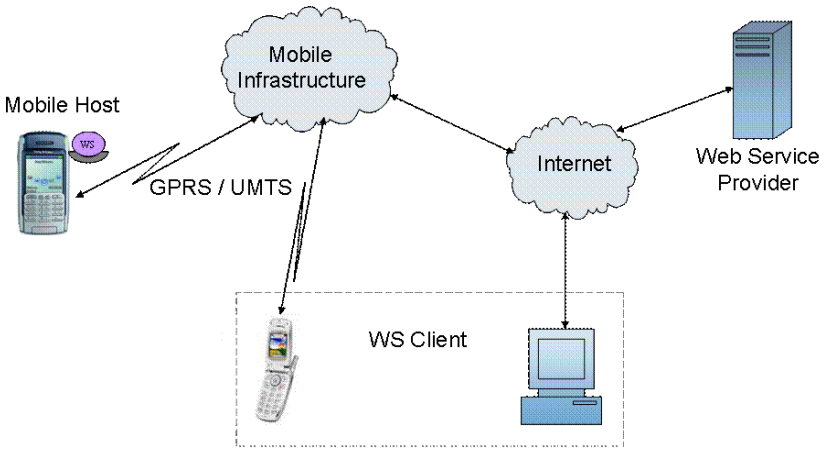


Figure 2.5: Mobile terminals as web service providers and clients (Adapted from [Srirama and Naumenko, 2007])

development for the smart phones.

Mobile terminals accessing the web services are common these days and mobile web service clients cater for anytime and anywhere access to services [Balani, 2003a; Forum Nokia, 2004; Ellis and Young, 2003; Benatallah and Maamar, 2003]. Some interesting mobile web service applications are the provisioning of services like e-mail, information search, language translation, company news etc., for employees who travel regularly. There are also many public web services accessible from smart phones like the weather forecast, stock quotes etc. Apart from the applications, there is also quite some research effort on efficient access of mobile web services, with minimum loads on resources of these mobile devices. For instance [Yang et al., 2003], propose an infrastructure for organizing and efficiently accessing mobile web services in broadcast environments. The idea with the approach is that, since sending data from a wireless device is the most power consuming process (in terms of battery life) broadcasting avoids this power consumption by avoiding the costly uplink transmissions [Imielinski et al., 1994]. The approach defines a multi-channel model to carry information about mobile web services. The *UDDI channel* includes registry information about m-services (mobile web services); the *m-service channel* contains the description and executable code of each mobile web service while the *data channel* contains the actual data needed for executing the mobile web service. The services are assumed to be within a given geographic area. The mobile service platforms are taking care of these resource issues, and the service platforms are discussed in the following subsection (section 2.3.1 on the facing page).

The usage of mobile terminals as web service client is also significant in the geospatial and location based services. Geographic Information Systems (GIS) make accessing geographical information service at anywhere and anytime feasible. Open Geospatial Consortium (OGC), which was previously known as Open GIS Consortium, is a non-profit, international, voluntary consensus standards organization that is leading the development

of standards for geospatial and location based services. As web service emerged as a new technology framework, OGC started designing and providing GIS service over Internet with web service standards [Whiteside, 2007]. OGC Web Services (OWS) represent evolutionary, standards-based frameworks that enable seamless integration of a variety of online geo-processing and location services. There is also significant work going on in accessing these OWS from smart phones and PDAs [Brisaboa et al., 2007].

Regarding industrial applications, Pulkkinen et al. have studied the business of remote maintenance services for machinery equipment in pulp&paper industry [Pulkkinen et al., 2007]. The case company, Metso Paper Inc. specializes in pulp and paper industry processes, machinery, equipment, related know-how, and after sales services. There is a need for the cross-organizational secure provisioning of maintenance services by experts who are usually on the move, and who have only their handheld devices, or laptops in the best case [Pulkkinen et al., 2007]. The maintenance experts use mobile phones, PDAs, laptops to access traditional web services that provide functionality for the condition monitoring, billing, maintenance, faults analysis, repair management, and other activities. These traditional web services are provided by machinery control systems or their vendors through Internet. A case study of one such a service could be found in [Johansson and Mollstedt, 2006]. Mobile phones host applications which are clients for these services. On the other hand, there is a need to collect data from mobile phones. These data includes history of maintenance activities (logs), collected information on customers' sites of different media (text notes, photos, videos, audio recordings, etc), current locations of experts, their availability (load of experts with other tasks and/or maintenance requests), and other. Mobile Web Service Provisioning (MWSP) (chapter 3 on page 35) may facilitate and generalize the design of solutions for accessing the data that reside on experts' mobile devices.

2.3.1 Platforms Supporting Mobile Web Services

Mobile applications unlike the normal desktop applications are particularly restricted by the runtime environment of the devices. Usually a mobile application can only run on certain models of mobile phones. The limitations come from different aspects: device operating system, the programming language and the platforms used to develop the application, device capabilities and the size of device storage. Therefore, many factors have to be considered for developing applications for the smart phones. Several software platforms like the PersonalJava, Symbian C++ etc. exist for the mobile phone application development. This subsection discusses some of these programming environments for smart phones.

Symbian OS (Operating System) C++

Symbian is an operating system derived from the Epoc operating system. Epoc was developed by Psion for their handhelds in the 80's [Harrison, 2003]. The C++ based

Symbian OS (Operating System) provides a secure, reliable operating system for mobile information devices. Being specifically designed for mobile devices, with low power consumption and small memory footprint, Symbian provides a stable platform for the telecommunications industry and technologies such as GPRS, Bluetooth, SyncML, and ultimately 3G. Symbian OS is not only an operating system but actually a full software and communications platform. The OS comes to real devices in different flavors. More precisely, Symbian Inc develops the base operating system and licenses it out to phone manufacturers. Vendors then build a user interface on top of the base operating system. The vendors can also customize the operating system for a specific purpose; the bundle of the operating system plus the user interface is shipped, with the hardware to be sold on the market.

Symbian OS phones available on the market are based on three user interfaces open to C++ programmers - *Nokia Series 80 Platform* [Forum Nokia, 2007c] (Nokia 9200 series communicator), *Nokia Series 60 Platform* [Forum Nokia, 2007b] (Nokia 7650, Nokia 3650) and *UIQ technology* (formerly known as User Interface Quartz) [UIQ, 2007] (SonyEricsson P800/P900/P910i/P990 smart phones). Apart from C++ support, all these designs are also open to Java programming. CodeWarrior for Symbian OS from Metrowerks, C++Builder Mobile Set from Borland and Visual Studio from Microsoft provide tool support for Symbian C++ programmers. Borland and Metrowerks are adding unique value in terms of integrated development environments, ease-of-use, debugging and support of symbian OS features.

PersonalJava

PersonalJava [Sun Microsystems, 2007e] also referred to as '*pJava*', is a Java programming environment targeted at developing applications for resource-constrained devices like smart phones, PDAs, set-top boxes and many other embedded devices. It was the first attempt by Sun to produce a Java application environment (JVM) for mobile devices. PersonalJava specifies a reduced set of class libraries compared to the Java desktop environment. Over the years, the Symbian OS port of PersonalJava [Symbian, 2007] has undergone substantial optimizations like assembler coded byte code interpreter, optimized AWT (Abstract Window Toolkit) library etc., that make PersonalJava in combination with the hardware performance enhancements, a powerful alternative for the development of mobile applications. In addition, the memory footprint of the PersonalJava application environment has been optimized to run in resource-limited environments while providing near desktop web-fidelity [Frank, 2004].

The PersonalJava profile is based on the JDK1.1 (Java Development Kit) [Sun Microsystems, 2008], but makes a number of packages, classes, and methods optional. It gives the capability to the developer to create Web applets and other mobile phone applications. PersonalJava predates the Connected Device Configuration (CDC) and is the forerunner to J2ME (Java 2 Platform, Micro Edition). Currently PersonalJava is going through *Sun End of Life (EOL)* process as J2ME is preferred ahead of the PersonalJava. Further support to

this platform will slowly be removed, even though some of today's smart phones support it. Some of the PersonalJava profile supported smart phones include Nokia Communicator 9200 series phone/PDAs (9210, 9290 and 9210i) and SonyEricsson P800/P900/P910i smart phones [Sun Microsystems, 2007c].

Java ME

Java^{TM} Platform, Micro Edition (Java ME) [Sun Microsystems, 2007g] previously known as *Java 2 Platform, Micro Edition (J2ME)* is a subset of the *Java 2 Platform, Standard Edition (J2SE)*, and is the most ubiquitous Java application platform for mobile devices, consumer and embedded devices such as mobile phones, PDAs, TV set-top boxes, printers, in-vehicle telematics systems, and a broad range of other embedded devices. The Java ME platform includes flexible user interfaces, a robust security model, a broad range of built-in network protocols, and extensive support for networked and offline applications that can be downloaded dynamically. Java ME is defined through the Java Community Process, and it also maintains the Java philosophy of portability. Java ME is very successful and every major manufacturer is embedding Java ME on some of their phones. An interesting note on J2ME is that all *Mobile Information Device Profile (MIDP)* implementations must provide support for the HTTP protocol [Fielding et al., 1999]. This guarantees the availability of HTTP as a transport mechanism for web services.

Java ME technology was originally created in order to deal with the constraints associated with building Java applications for small device with limited memory, display and power capacity. Java ME platform is a collection of technologies and specifications that can be combined to construct a complete Java runtime environment specifically to fit the requirements of a particular device or market. The Java ME technology is based on three elements; *configuration* that provides the most basic set of libraries and virtual machine capabilities for a broad range of devices, *profile* as a set of APIs that support a narrower range of devices and an *optional package* as a set of technology-specific APIs.

Over time the Java ME platform has been divided into two configurations, a vertical sets of virtual machines and minimal set of class libraries providing the base set of functionality for the range of devices within each configuration. The two configurations are the *Connected Limited Device Configuration (CLDC)* [Sun Microsystems, 2000] and the *Connected Device Configuration (CDC)* [Sun Microsystems, 2005]. The CLDC is the smaller of the two platforms, and caters for devices such as mobile phones with an intermittent network connection, slow processor (16 or 32 bit), and limited memory (128kb-512kb). The CDC is designed for devices such as high end PDAs, set-top boxes and communicators, with more memory (minimum of 2Mb) and a 32-bit processor.

CLDC is specifically designed to meet the needs for a Java platform to run on devices with very limited memory, processing power and graphical capabilities. The development environment for Java ME on the CLDC devices is the Mobile Information Device Profile (MIDP). It defines the classes for user interface, persistent storage, and networking and application management. Combined with CLDC and its *Kilo Virtual Machine (kVM)* [Sun

Microsystems, 2000], this profile provides a complete Java runtime environment (JRE) for mobile phones and devices with similar capabilities. A MIDlet (Mobile Informational Device Application) is the application written by software developer, such as a game or a business application, created for MIDP. These MIDlets can be downloaded over the air and installed on the mobile phones.

CDC comes with three different profiles; the *Foundation Profile* (Java Specification Request (JSR) 219), the *Personal Basis Profile* (JSR 217) and the *Personal Profile* (JSR 216). Out of the three profiles Personal Profile is aimed at devices that require full Graphical User Interface (GUI) or Internet applet support, such as high-end PDAs, communicator -type devices, and game consoles. It includes the full Java Abstract Window Toolkit (AWT) libraries and offers Web fidelity, easily running Web-based applets designed for use in the desktop environment. Personal Profile replaces PersonalJava technology, and provides PersonalJava applications a clear migration path to the Java ME platform.

Combining various optional packages can further extend the Java ME platform. These optional packages along with CLDC, CDC, and their corresponding profiles, can address very specific market requirements. These optional packages offer standard APIs to support both existing and emerging technologies like Bluetooth, web services, wireless messaging, multimedia, and database connectivity etc.

.NET Compact Framework

Microsoft .NET Framework [MSDN, 2007b] is a software component added to the Microsoft Windows operating system. The .NET Compact Framework [MSDN, 2007a] is a subset of .NET Framework and provides a robust environment for developing mobile applications. It provides a large body of pre-coded solutions to common program requirements, and manages the execution of programs written specifically for the framework. The framework is intended to be used by applications created for the Windows platform. The .NET Compact Framework's managed code and web services enable the development of secure, downloadable applications on devices such as personal digital assistants (PDAs), mobile phones, and set-top boxes. The framework uses some of the same class libraries as the full .NET Framework and also a few libraries designed specifically for mobile devices. The .NET Compact Framework is more successful in the PDA market, where Windows enjoys some success rather than in smart phones [Elkarra, 2003].

2.3.2 SOAP Implementations for Resource Constrained Environments

Apart from the runtime environment of the devices, for the mobile phone to act as a web service client, the smart phone should be able to consume web service messages. To request a web service, the smart phone should create the web service request messages (SOAP requests), send it to the web service provider and be able to process the response messages back from the provider. For this the smart phone should support a SOAP parser

for processing the web service messages. Many SOAP parsers like kSOAP2, WSOAP etc. exist specifically for the resource constrained smart phones. This subsection discusses some of these platforms and SOAP processors.

gSOAP

The gSOAP toolkit is a platform-independent development environment for C and C++ web services [van Engelen and Gallivan, 2002]. gSOAP provides a 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. The gSOAP toolkit is a mature and fast toolkit and is available in open source form. The toolkit supports many platforms, including embedded systems and follows the WS-I Basic Profile 1.0a compliance recommendations.

eSOAP

Embedded SOAP (eSOAP) [Silva, 2001; da Silva, 2007] is a small lightweight implementation of the SOAP 1.1 specification and is exclusively designed for embedded systems. The eSOAP toolkit has C++ and Java libraries that provide a SOAP processing engine for the embedded system. eSOAP achieves easy interoperability for networked embedded systems and the library is compact with a memory footprint less than 150KB (Kilobyte). The toolkit is also portable as the core engine of eSOAP is totally written in ANSI (American National Standards Institute) C++. It uses C++ Standard Template Library (STL) whenever possible and where STL is not available, it provides a container library. The *esoapcg* compiler provided with the toolkit generates C++ mappings to and from WSDL and SOAP. The complier generates a C++ stub and proxy for a given web service interface. The toolkit also provides a Java library with interface and classes for web service client development.

Wireless SOAP (WSOAP)

The Wireless SOAP (WSOAP) [Apte et al., 2005] aims to provide static encoding based on SOAP schema, leverages WSDL service description to create adaptive encoding for web service interfaces. WSOAP is actually a set of optimization techniques. The approach concentrates on functional message equivalence also called Name Space Equivalency 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 size by 3-12 times. WSOAP limits computational cost by favoring lookup over computation wherever possible.

Wingfoot SOAP

Wingfoot SOAP [Wingfoot, 2007b] is a lightweight client implementation of SOAP 1.1. It is specifically targeted at the MIDP/CLDC platform but can also be used in PersonalJava, J2SE and J2EE (Java 2 Platform, Enterprise Edition) environments. Wingfoot SOAP provides two different binaries: `kvmwsoap_1.06.jar` targeted at the CLDC/MIDP platforms. The binary package includes a lightweight XML parser and is 37K in size. The XML parser is based on kXML. The second version `j2sewsoap_1.06.jar` targets at the CDC/Personal Java, J2SE, and J2EE platforms. It also includes the lightweight XML parser and is 34.5K in size [Wingfoot, 2007a]. The API for both binaries is identical except for the Transport implementation. Wingfoot SOAP provides its own mechanism of sending the SOAP messages over HTTP. The toolkit provides two implementation of Transport: `HTTPTransport` provides a means for MIDP applications to send a SOAP payload over HTTP; `J2SEHTTPTransport` provides PersonalJava/CDC, J2SE and J2EE applications to send a SOAP payload over HTTP. SOAP over alternative transport layers like SMTP (Simple Mail Transfer Protocol), UDP (User Datagram Protocol) etc. can be realized using Wingfoot SOAP, by implementing the Transport interface.

Java Specification Request 172 (JSR 172)

The Java Specification Request 172 (JSR 172) [Ellis and Young, 2003], defined by the Java Community Process (JCP), defines the J2ME web services specification and a set of API for accessing web services from the J2ME environment. Sun also has provided a reference implementation of this API with the J2ME Web Services APIs (WSA) [Sun Microsystems, 2007a]. WSA enables J2ME devices to be web services clients, providing a programming model that is consistent with the standard web services platform. The API facilitates web service access from both CDC and CLDC configurations. It provides a stub compiler that generates the code needed by the J2ME applications to execute simple SOAP calls to an existing web service. JSR 172 also provides a light weight XML parser for the J2ME platform. JSR 172 is the first attempt to get standardization into mobile web services.

kSOAP2

kSOAP is an open source API for SOAP parsing [kSOAP, 2007]. It is based on kXML parser, a lightweight, open source XML parser. kXML uses *XML pull parser* mechanism a slight modification of *DOM (Document Object Model) parser* [Balani, 2003b]. Using the pull parser, the application is in control of when and where it asks the parser for the next event. The advantage is that the processing state can be implemented much more natural in local variables and recursions. If we want to parse a small fragment of data, may be at the middle of the document, the data already parsed need not reside in memory, in contrast to DOM, where all the data resides in memory.

kSOAP provides a SOAP parser with special type mapping and marshalling mechanisms. Both kSOAP and kXML are thin, easy to use, and well documented, and hence can be used for resource-constrained devices like mobile phones. The kSOAP parser understands the data-type information in SOAP messages and automatically converts the SOAP message to Java data objects, similar to SOAP parsers. The parser provides programming transparency between a Java program and a SOAP message. A programmer just feeds Java objects into a SOAP writer, sends the message, waits for the server response, and then reads Java objects directly from the SOAP parser. kSOAP is deprecated and completely redesigned to kSOAP2 [kSOAP2, 2007]. kSOAP2 has improved support for literal encoding and made *SOAP Serialization* support to be optional. kSOAP2 uses kXML2, the updated version of kXML and the kSOAP2 improvements require some additional effort when porting applications from kSOAP to kSOAP 2.

Table 2.1 summarizes the SOAP platform and implementation supports available/adaptable for smart phones. Since the thesis mainly considered the Java platforms (J2ME / PersonalJava) and provisioning of service from smart phones (chapter 3 on page 35), it used kSOAP2 for the SOAP processing. Moreover kXML and kSOAP were one of the first XML and SOAP processors, that can be adapted for resource constrained smart phones.

SOAP Engine	**Programming Language**	**Supported Platforms**	**Standards Compliance**
gSOAP (Open source)	C, C++	Embedded systems, Symbian	WS-I Basic Profile 1.0a
eSOAP	C++ Java	Embedded systems	SOAP 1.1 SOAP 1.1 (only client)
WSOAP	Java	J2SE	Proprietary SOAP
Wingfoot SOAP	Java	MIDP/CLDC J2EE, PersonalJava, CDC	SOAP 1.1
JSR 172 (JCP)	Java	J2ME CDC/CLDC	SOAP 1.2 (only client)
kSOAP / kSOAP2 (Open source)	Java	J2ME PersonalJava	SOAP 1.1 / SOAP 1.2

Table 2.1: SOAP engines supporting mobile web services

2.3.3 Standardization Efforts for Mobile Web Services

Previous subsections have discussed a wide range of platforms, protocols, SOAP engines and implementations, supporting the development of mobile web services. Most of them are proprietary and have evolved concurrently. Hence there are some standardization efforts from different groups to achieve some uniformity among these developments. This subsection discusses some of these prominent efforts.

JSR 172

There is currently no standardized web services support for mobile environments. JSR 172 is the first attempt to get standardization into mobile web services [Ellis and Young, 2003]. JSR 172 defines a J2ME web services specification and thus extends the web services platform to include Java ME client devices. J2ME web services specification defines two new optional packages: *XML Processing APIs* and *RPC-based access to web services*. JSR 172 reference implementation by Sun also provides a stub compiler that generates the code needed by the J2ME applications to execute simple SOAP calls to an existing web service. The web service client applications are portable and the specification allows generated stubs to be independent of implementation, so that these applications can be dynamically provisioned to any J2ME technology-supported platform. With JSR 172, J2ME client communication with web services has gained a programming model that is consistent with its counterparts for other clients, such as Java 2 Platform, Standard Edition (J2SE^{TM}) technology. Services and clients no longer have to be implemented on the same platform or by the same organization [Sun Microsystems, 2007d].

LA

Liberty Alliance (LA) project is the 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. The members of the Liberty Alliance envision a networked world across which individuals and businesses can engage in virtually any transaction without compromising the privacy and security of vital identity information [LA, 2007]. The Liberty Alliance project proposes the use of federated network identity to solve the problems of network identity. 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 *Liberty Identity Web Services Framework (ID-WSF)* [Tourzan and Koga, 2006] builds upon the federated identity foundation and provides a framework for identity-based web services in a federated network identity environment. ID-WSF defines a SOAP based invocation framework with a layered architecture. The framework does not specify any contents for the SOAP body, allowing the development of identity services within the context of the Liberty Identity Web Services Framework. Nokia has developed *Nokia Mobile Web Services Framework*

for its S60 and Series 80 phones, based on this Liberty ID-WSF specification [Forum Nokia, 2007a].

OMA

Open Mobile Alliance (OMA) group is directed at defining 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 [OMA, 2004]. Mobility and roaming are the obvious key characteristics which are hindrances to mobile web service interactions. The current possible mobile web service applications have a number of drawbacks. 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 Wireless Application Protocol (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* [OMA, 2006b] and *OMA Mobile Web Services Requirements specification* [OMA, 2006a] are 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)

2.3.4 Devices and Platforms Considered in the Thesis

Any mobile phone built in with the platforms, discussed in section 2.3.1 on page 19, can support mobile web service development. Nokia series 60 and series 80 phones, Sony Ericsson UIQ series smart phones and Siemens Sxx series phones are among the prominent devices mainly used in mobile web services development. Considering the smart phones we possessed, SonyEricsson P800, P910i, W810i, P990i and Nokia N70, and the ease of programming and the supported platforms and tools for the test devices, we have considered the Java environment for the mobile web services development. Some of the analysis considered in this thesis targeted PersonalJava platform and others are based on J2ME. Table 2.2 on the next page provides the platforms and APIs supported by these smart phones [Sun Microsystems, 2007c].

Mobile phone	Supported platforms	Supported API
Sony Ericsson P800	PersonalJava 1.1.1, CLDC 1.0, MIDP 1.0, Symbian OS, UIQ	
P910i	PersonalJava 1.1.1, CLDC 1.0, MIDP 2.0, Symbian OS, UIQ	MMAPI (Mobile Media Application Programming Interface) / JSR 135, WMA (Wireless Messaging API) 2.0 / JSR 205
P990i	CDC 1.0, CLDC 1.1, Foundation Profile, MIDP 2.0, Personal Profile, Symbian OS, UIQ	MMAPI, WMA 2.0, Web Services (JSR 172)
W810i	CLDC 1.1, MIDP 2.0	MMAPI, WMA 2.0, Web Services (JSR 172)
Nokia N70	CLDC 1.1, MIDP 2.0	MMAPI, WMA 2.0, Web Services (JSR 172), Nokia UI (User Interface) API

Table 2.2: Platforms and APIs supported by the smart phones considered in the thesis

2.4 SOAP over Different Transportation Protocols

SOAP messages can be carried on top of any underlying protocols such as HTTP, TCP (Transmission Control Protocol), UDP (User Datagram Protocol), BEEP (Block Extensible Exchange Protocol) and SMTP (Simple Mail Transfer Protocol). Thus, SOAP creators have defined a binding framework for SOAP instead of a fixed binding. Specifically, the *SOAP binding framework specification* [Gudgin et al., 2003] provides a high level of flexibility in terms of how SOAP messages are transmitted. The following subsections introduce SOAP over different protocols, and address the research going on in this domain.

2.4.1 SOAP over HTTP

HTTP protocol is the most widely used transport mechanism for SOAP binding, among the web services community. SOAP over HTTP is also the only concrete binding specification defined in the SOAP binding framework proposal. There are many reasons why HTTP is an attractive binding option. HTTP is already widely used on the Internet and universally supported by web servers. This provides a strong base for the adoption of web services with SOAP over HTTP binding. Moreover HTTP uses TCP/IP (Transmission Control Protocol/Internet Protocol) as its underlying transport which ensures packets are reliably

Listing 2.1: SOAP request message over HTTP transportation protocol

```
POST / HTTP/1.1
SOAPAction: expertSearch
Content-Type: text/xml
Content-Length: 1022
User-Agent: kSOAP/2.0
User-Agent: UNTRUSTED/1.0
Host: 187.226.82.94:8668

<soap-env:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap-enc="http://www.w3.org/2001/12/soap-encoding"
  xmlns:soap-env="http://www.w3.org/2001/12/soap-envelope">
  <soap-env:Header/>
  <soap-env:Body>
    <n0:expertSearch

    . . .
    </n0:expertSearch>
  </soap-env:Body>
</soap-env:Envelope>
```

delivered and most firewalls allow HTTP packets to pass through them.

SOAP over HTTP message can be transported by encapsulating the SOAP request into the HTTP GET or POST message body. Similarly, a SOAP response can be encapsulated into the body of the HTTP response. The message formats of the HTTP request and response are described in Appendix A on page 189. Listing 2.1 and 2.2 on the next page show the SOAP request and response messages transmitted over the HTTP protocol. The service being invoked (`expertSearch`) is to find an expert on a specific topic in an m-learning community. The service is explained in detail in chapter 7 on page 161.

2.4.2 SOAP over Alternative Transportation Protocols

Instead of transmitting SOAP over HTTP, the message can directly be transported using TCP as the underlying protocol. The SOAP messages are contained into the data octets of a TCP packet. There is not yet an official specification for *SOAP binding with TCP*; however, Apache Axis2 [Apache Software Foundation, 2007a] and Microsoft WSE (Web Service Enhancement) 2.0 [MSDN, 2007c] already include APIs that enable sending SOAP messages via TCP channel. Axis2 also supports *SOAP over JMS*. JMS (Java Message Service) offers Java programmers a common way to create, send, receive and read enterprise messages. JMS is appropriate transport protocol for SOAP when there is a requirement for web services to communicate asynchronously and reliably. Asynchronicity ensures the sender of a message does not have to wait for a reply to the message and reliably assures the sender that the message will be delivered. But the problem with

Listing 2.2: Valid SOAP response message over HTTP transportation protocol

```
HTTP/1.0 200 OK
Server: MobileHost
Date: Fri Aug 24 12:21:15 UTC 2007
Content-Length: 585
Last Modified: Fri Aug 24 12:21:15 UTC 2007
Content-type: text/xml
Request-ID: 3
Connection: close

<soap-env:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:soap-enc="http://www.w3.org/2001/12/soap-encoding"
   xmlns:soap-env="http://www.w3.org/2001/12/soap-envelope">
   <soap-env:Header/>
   <soap-env:Body>
     <n0:expertSearchResponse

     . . .
     </n0:expertSearchResponse>
   </soap-env:Body>
</soap-env:Envelope>
```

current reliable message transport mechanisms is that they require communicating parties to be using the same infrastructure of JMS such as IBM WebSphere MQ [IBM Corporation, 2008]. Considering this issue, WS-ReliableMessaging [Bilorusets et al., 2005] draft standard has been developed to provide a framework for interoperability between different reliable transport infrastructures.

SOAP over email binding (SMTP) is also possible and is presented in the W3C specification. However SOAP over SMTP is only presented in the specification as an example to demonstrate the realization of the SOAP binding framework. In the SOAP over SMTP, SOAP messages are encapsulated in the bodies of emails. SOAP over SMTP only allows asynchronous message exchange between web services [Cunnings et al., 2001]. *SOAP over BEEP* (Block Extensible Exchange Protocol) is also possible and is studied by [Kefali, 2004]. *SOAP over JXTA*, a peer to peer (P2P) technology, is addressed in section 3.5.4 on page 64. Here the SOAP request/response messages are exchanged over the JXTA pipes.

Apart from these methods, *SOAP over UDP* specification [Gudgin et al., 2004] defines the means of encapsulating SOAP messages into data octets part of a UDP packet. With this specification the SOAP message must be small enough to fit in one UDP packet. So the maximum size of the message should be 65,536 bytes (2^{16}). The specification supports four messaging patters; *Unicast one-way*, *Multicast one-way*, *Unicast request, unicast response* and *Multicast request, unicast response*. Lai et al. argue through extensive set of experiments that the throughput of SOAP over UDP is 10 times higher than SOAP

over HTTP in Wi-Fi environments [Lai et al., 2005]. However UDP is highly unreliable and packets delivered by UDP may be duplicated, arrive out of sequence or even not reach their destination at all. But [Aijaz, 2006] suggests that UDP can be used to reliably transport messages by applying a simple Automatic Repeat Request (ARQ) protocol over UDP. The approach distinguishes the cases of reliable and unreliable binding and presents a system for an unreliable and reliable UDP SOAP binding for a mobile web service based middleware [Gehlen et al., 2006; Gehlen, 2007].

Considering the issues discussed above and not having proper standards and specifications, the thesis proceeded with SOAP over HTTP for the Mobile Host (section 3.1 on page 35) implementation. Moreover J2ME, the current defacto standard in mobile application development, mandates that all Mobile Information Device Profile (MIDP) implementations must provide support for the HTTP protocol. This guarantees the availability of HTTP as a transport mechanism for mobile web services. If any of the transport mechanisms is standardized and the web service community adapts the mechanism, Mobile Host can later be adapted to this framework. The only change would then be at transport binding. But considering the current trend in mobile web services development, this might take some time to materialize.

2.5 Telecommunication Technologies

This subsection introduces some of the telecommunication technologies and developments relevant for this thesis. The telecommunication systems have evolved across time. The *First generation* telecommunication technologies were analog systems where voice was considered to be the main traffic. The *Second generation* (2G) Global System for Mobile Communications (GSM) [Redl et al., 1995] networks deliver high quality and secure mobile voice and data services like SMS, circuit switched Internet access etc., with full roaming capabilities and across the world. The GSM platform is a widely successful wireless technology and is still the world's leading mobile standard. Some of the well known second generation standards relevant for the thesis are GSM, CSD, HSCSD. High Speed Circuit Switched Data (HSCSD) [Capone and Musumeci, 1997] is an enhancement of CSD (Circuit Switched Data) data services of current GSM networks. HSCSD allows the access of non-voice services with about 3 times higher data rates than CSD. The higher rates are enabled by using multiple channels for the data transmission. The HSCSD allows access to company LANs (Local Area Network), send and receive e-mails, access the Internet, etc. With this technology, subscribers are able to send and receive data from their portable computers or mobile devices at a speed of up to 28.8 kbps.

Interim generation (2.5G) is a transition between 2G and 3G cellular wireless technologies. The most common technology in this generation is the General Packet Radio Service (GPRS) [Rysavy, 1998; ETSI, 1997]. GPRS is an extension of second generation GSM and provides short connection setup times and packet switched connections. GPRS

offers faster data transmissions, up to 115Kbps via a GSM network. The high bandwidth is achieved by combining up to eight time slots at the radio interface, where the data is transported in a packet-oriented way. The available bandwidth can be shared among different users. Hence the total available bandwidth can be immediately dedicated to those users who are actually sending at any given moment. Another major benefit of GPRS is with its billing per kilobyte of information transmitted, where as in CSD, connections are billed per second.

To meet the growing demands with the increase in number of subscribers, and the transmission rates required for high speed data transfer and multimedia applications, *Third generation* (3G) telecommunication technologies and standards have evolved. Most prominent 3G systems are Universal Mobile Telecommunications System (UMTS) and Freedom of Mobile Multimedia Access (FOMA). UMTS [Umtsworld, 2002] generally uses W-CDMA (Wideband Code Division Multiple Access) as the underlying air interface and is standardized by the 3GPP (Third Generation Partnership Project) [3GPP, 2007]. FOMA is the brand name for the 3G services being offered by NTT DoCoMo [NTT DoCoMo, 2007a]. The systems in this standard are basically a linear enhancement of 2G systems. The greater network capacity in 3G is achieved through improved *spectral efficiency*. 3G offers a transmission rate of 384 Kbps for mobile systems and 2 Mbps (Megabytes per second) for stationary systems.

Further developments in cellular world are leading towards the next generation of telecommunication technologies. *Fourth generation* Communications System (4G) will provide an end-to-end IP solution where voice, data and streamed multimedia can be served to users on an "Anytime, Anywhere" basis. A large amount of the work is aimed at simplifying the architecture of the current 3G systems, which is based on two parallel infrastructures consisting of circuit switched and packet switched network nodes respectively. 4G will allow only packet switching. 3GPP LTE (Long Term Evolution) is the project within the 3GPP to improve the UMTS standard to cope with these future requirements. 4G will be capable of providing 100 Mbps and 1 Gbps, in outdoor and indoor environments, respectively with end-to-end quality of service and high security [Kim and Prasad, 2006]. NTT DoCoMo had already achieved 2.5Gbps packet transmission in the downlink with 4G, while moving at 20km/h [4GPress, 2005].

4G will be based on all-IP packet switching only. So, all the systems and wireless devices connected to the network are to be addressed with IP address. Hence IPv4, the current version of the Internet Protocol, for general use on the Internet is not sufficient, which has limited set of public addresses. Internet Protocol version 6 (IPv6) [Johnson et al., 2004] has evolved as the new network layer protocol for packet-switched internetworks. IPv6 has a much larger address space and is able to support 2^{128} (about $3.4*10^{38}$) addresses. By increasing the number of IP addresses, IPv6 removes the need for Network Address Translation (NAT), used for sharing a limited number of addresses among a larger group of devices.

2.6 Summary

This chapter discussed the state of the art and motivation for the research addressed by the thesis. The chapter first introduced the web services technology along with associated standards and protocols. Later it introduced the wireless developments in terms of device and transmission capabilities. The developments in these two domains lead to a new domain of applications, mobile web services. The chapter introduced the supported devices and platforms, standardization efforts and implementations, and SOAP transmission mechanisms for mobile web services. The chapter concluded with a discussion of the current generation telecommunication technologies.

3 Mobile Web Service Provisioning

While mobile web service clients are common these days, and many software tools already exist in the market, easing their development and adoption, as discussed in section 2.3 on page 17, the research with providing web services from smart phones is still sparse. However, a mobile device in the role of a service provider enables, amongst others, entirely new scenarios and end-user services. This chapter explains the research with mobile web service provisioning and tries to provide motivation and state of the art for the upcoming chapters.

3.1 Mobile Host

The paradigm shift from the role of service consumer to the service provider is a step towards practical realization of various computing paradigms such as pervasive computing, ubiquitous computing, ambient computing and context-aware computing. For example, the applications hosted on a mobile device provide information about the associated user (e.g. location, agenda) as well as the surrounding environment (e.g. signal strength, bandwidth). Mobile devices also support multiple integrated devices (e.g. camera) and auxiliary devices (e.g. GPS (Global Positioning System) receivers, printers). For the hosted services, it provides a gateway to make available its functionality to the outside world (e.g. providing paramedics assistance). These developments have resulted in the research paradigm which is referred to as *Nomadic Mobile Services Provisioning*. A Nomadic Mobile Service (NMS) is hosted by the mobile host such as a handheld device, mobile phone or an embedded device capable of connecting to the Internet using a wireless network. The mobile device roams from one mobile network to another which gives nomadic characteristics to the services it hosts.

Nomadic Mobile Services Provisioning was attempted with different proprietary technologies, just like mobile services from section 2.3 on page 17. For instance, van. Halteren et. al. proposes a proxy based middleware based on the Jini surrogate architecture [van Halteren and Pawar, 2006]. This Jini based nomadic mobile services provisioning approach is explained in section 3.2 on page 46. Once again these proprietary approaches seriously affected the interoperability of the provided NMS. So to overcome this uninteroperability and the integration of NMS issues, in the *mobile web service provisioning* project [Srirama, 2007], a web services based *Mobile Host* was developed and its performance was extensively analyzed, proving the feasibility of concept [Srirama, 2004].

The main benefit of Mobile Host is the achieved integration and interoperability for

the mobile devices. It allows applications written in different languages and deployed on different platforms to communicate with Mobile Host over the cellular network. Without the mobile web service provisioning approach, to send a message to the mobile phone, WAP push protocol is to be used. WAP push enables applications to alert the users with information like time and location sensitive information. For achieving WAP 1.x push, the Push Initiator sends the content to the Push Proxy Gateway (PPG) using the Push Access Protocol (PAP). The PPG attempts to find the correct destination of the mobile terminal and delivers the push submission to the mobile using the Push Over-the-Air protocol. For establishing this connection, the PPG sends a SMS message with Session Initiation Request (SIR) to the mobile terminal. The terminal responds by initiating appropriate bearer and contacts the PPG. WAP 2.0 assumes a HTTP server on the phone, if the server doesn't exist; it follows the WAP 1.x mechanism [Pashtan, 2005]. But the approach is purely WAP specific. Now with mobile web service provisioning approach the content can be sent directly to the mobile by initiating a web service request to the mobile terminal. Moreover by following the web services approach the Mobile Host benefits from the open standards, widely accepted protocols and extensive development support. The following subsections explain the Mobile Host's architecture, implementation details and the performance analysis.

3.1.1 Architecture and Implementation Details of the Mobile Host

Mobile Host is a light weight web service provider built for resource constrained devices like cellular phones, developed in collaboration with Ericsson, during the author's Master thesis [Srirama et al., 2006d; Srirama, 2004]. Figure 3.1 on the facing page shows the basic mobile web services framework with web services being provided from the Mobile Host. Though Mobile Host was developed for resource constrained devices, it followed the same architecture as general web services, with proper optimizations wherever necessary. 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 over different transportation protocols like HTTP, BEEP, UDP, and WAP etc. In the Mobile Host's implementation, the web service requests sent by HTTP tunneling are diverted and handled by the web service handler. SOAP over alternate protocols to HTTP and alternate approaches for the Mobile Host development over these transportation protocols are addressed in section 2.4.2 on page 29. 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 3.2 on page 38 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

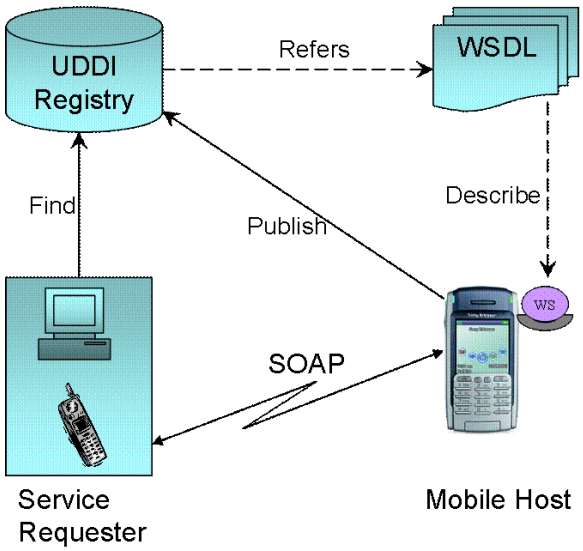


Figure 3.1: Basic mobile web services framework with the Mobile Host (Adapted from [Srirama et al., 2007a])

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.

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 the actual logic of the service. 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 initial Mobile Host was developed in PersonalJava [Section 2.3.1] on a SonyEricsson P800 smart phone. The footprint of the fully functional prototype is only 130 KB. Open source kSOAP2 [Section 2.3.2] 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

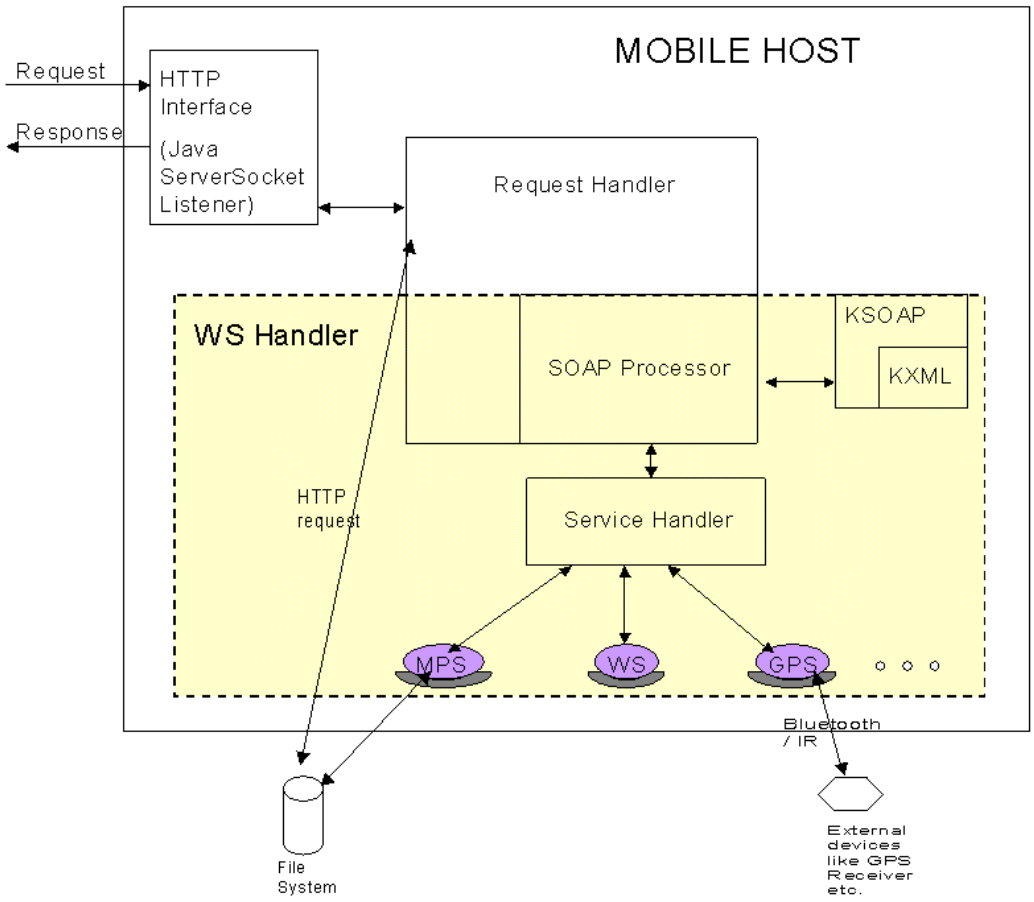


Figure 3.2: Core architecture of the Mobile Host (Adapted from [Srirama et al., 2006d])

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. Along with these basic features, a light weight Graphic User Interface (GUI) was provided to activate and deactivate the deployed services as and when necessary, so as to control the load on the Mobile Host. The GUI also has support for providing memory usage details of the smart phone and the basic server operations like start, stop and exit, thus helping in evaluating the performance analysis of the Mobile Host.

Similar implementations of Mobile Host are also possible with other Java variants like J2ME [Section 2.3.1], for smart phones [Gehlen, 2007]. This study also has developed a J2ME based Mobile Host and its performance was observed to be not so significantly different from that of the PersonalJava version. The detailed performance evaluation of the J2ME based Mobile Host is addressed in chapter 4 on page 77, while discussing the

scalability analysis of the Mobile Host.

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 studied, where the thesis tried to realize Mobile Host in a Peer to Peer (P2P) [Shirky et al., 2001] network, there by leveraging the advertising and searching of WSDL documents to the P2P network. The detailed explanation of this approach is addressed in chapter 5 on page 119, 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 in gSOAP, eSOAP etc [Section 2.3.2]. But while developing the Mobile Host only the basic mobile web services architecture was considered, as the main interest was to check the feasibility with performance, of having such a standard Mobile Host on smart phone. The thesis considered this option of using SOAP proxy in the later study, to help in providing better QoS and discovery mechanisms for the Mobile Host. The details of the proxy architecture are addressed, while discussing the mediation framework in chapter 6 on page 145.

3.1.2 Sample Web Services Provided by Mobile Host

Before considering the Mobile Host's performance analysis and applications, this subsection describes some of the basic web services provided from the Mobile Host. The services give an idea of some of the services possible from smart phones. These services were used in calculating the performance loads of the Mobile Hosts.

Mobile Photo Album Service (Mobile Picture Service)

Generally, today's smart phones are 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) [Novak and Svensson, 2001] 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 (Location Information Service)

This dedicated web service provides the exact location information of the mobile terminal, such as *Global Positioning System (GPS)* data [NAVSTAR, 1995]. The GPS is a Department of Defense (DoD) developed, worldwide, satellite-based radio navigation system. The constellation consists of 24 operational satellites and is fully operational since 1995. GPS provides two levels of service, *Standard Positioning Service* and the *Precise Positioning Service*. The Standard Positioning Service is a positioning and timing service which will be available to all GPS users on a continuous, worldwide basis with no direct charge. The Precise Positioning Service is a highly accurate military positioning, velocity and timing service which will be available on a continuous, worldwide basis to users authorized by the U.S. [NAVSTAR, 2007].

The Location Data Provisioning Service uses a *Socket GPS receiver* for getting the GPS co-ordinates using Standard Positioning Service. 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 traveler's diary etc. The GPS co-ordinates can always be mapped to geo spatial maps.

3.1.3 Performance Evaluation of the initial 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 information service and some more basic services like echo, ls services and etc. The test setup comprised a Mobile Host developed and deployed on the SonyEricsson P800 smart phone and a standalone Apache Axis [Apache Software Foundation, 2007d] web service client. The smart phone had an internal memory of 12 Mb and a 16 MB (Megabyte) memory stick duo card. The ARM9 processor of the device clocked at 156MHz. The Axis client invoked different services (Within the context of this discussion, it is assumed that the client knows the exact location (Uniform Resource Identifier (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. Detailed description of mobile terminal access in different environments is addressed in chapter 5, while discussing Mobile Terminal Access in P2P networks (section 5.2).

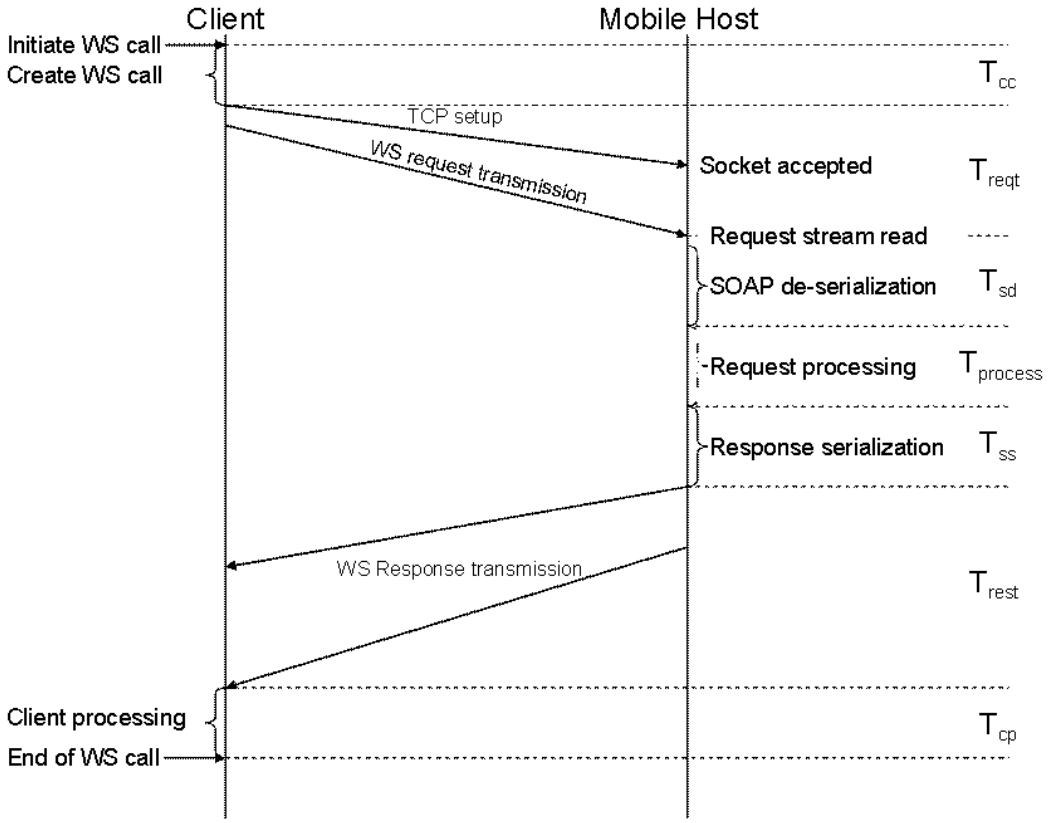


Figure 3.3: Mobile web service invocation: Operations and time stamps (Adapted from [Srirama et al., 2006a])

Mobile Host's Performance Model

Figure 3.3 shows different operations performed and time components that constitute one complete mobile web service invocation cycle, along the time axes. 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. The total time taken for this mobile web service invocation (T_{mwsp}) constitutes, the time taken by client for constructing valid SOAP message (T_{cc}), the time taken to transmit the SOAP request to Mobile Host (T_{reqt}), the time taken for de-serializing the XML based SOAP message to SOAPEnvelope object (T_{sd}), the time taken by the Mobile Host to execute the respective business logic and to populate the response ($T_{process}$), the time taken for serializing the SOAPEnvelope object back to XML data streams (T_{ss}), the time taken to transmit the SOAP response back to the client (T_{rest}) and lastly the time taken by the client to process the response (T_{cp}). The invocation process is shown in figure 3.3 and the total time taken for the mobile web service invocation is given in the following equation.

$$T_{mwsp} = T_{cc} + T_{reqt} + T_{sd} + T_{process} + T_{ss} + T_{rest} + T_{cp} \tag{3.1}$$

The request and response messages are transferred to the Mobile Host in the form of TCP packets. So some delay could be caused by packet loss, TCP acknowledgements, TCP congestion control etc. The delay is shown in the figure as the slanting lines for request and response transmissions. T_{tcp} represents this delay caused by the transmission protocol.

$$T_{tcp} = \delta_{reqt} + \delta_{rest} \tag{3.2}$$

where δ_{reqt}, δ_{rest} are the respective propagation delays caused while transmitting the SOAP request and response messages. In the proposed performance model the transmission times (T_{reqt}, T_{rest}) also include these tcp delays and an estimation of the tcp delays is not specifically observed. Moreover, the exact estimation of the T_{reqt} and T_{rest} times is not possible as their calculation process needs the synchronization of time stamps of both Mobile Host and mobile web service client, at the kVM level. Hence only the total transmission delay (T_{trans}) can be calculated. Besides, T_{cc} and T_{cp} are almost negligible, as the client in the analysis is a PC. Hence T_{trans} can be obtained by subtracting the total server processing time from the T_{mwsp}, as shown in the following equation.

$$T_{trans} \approx T_{mwsp} - T_{sd} - T_{process} - T_{ss} \tag{3.3}$$

Test Case Services Considered in the Analysis

For the analysis, as test cases for the mobile photo album service, 15 different images were selected with memory sizes ranging from 3Kb to 100Kb. The web service client tried to browse through these pictures. The location (GPS) data provisioning service used an external GPS device for providing the GPS data to the client and had a response size of approximately 2Kb. Alternatively, the pictures were directly taken by the smart phone and were provided to the client along with the location information. In the latter approach both the picture and the location details were invoked in a single service call, checking the effects of composite web services on the performance of Mobile Host. The size of the picture, directly taken by the smart phone was configured to have a size of approximately 40Kb.

The services, mobile photo album service and location (GPS) data provisioning service, were also observed to be quite relevant for the performance evaluation. The response of the mobile picture service is comparatively large (approximately 40KB) and this gave a large scope for observing the effects of different parameters like transmission delays, the encoding performed on the response messages, the actual service delay, and etc. on the performance of Mobile Host. The location information service returns just a small string (approximately 2 KB) containing the GPS data as the response, which gave the scope

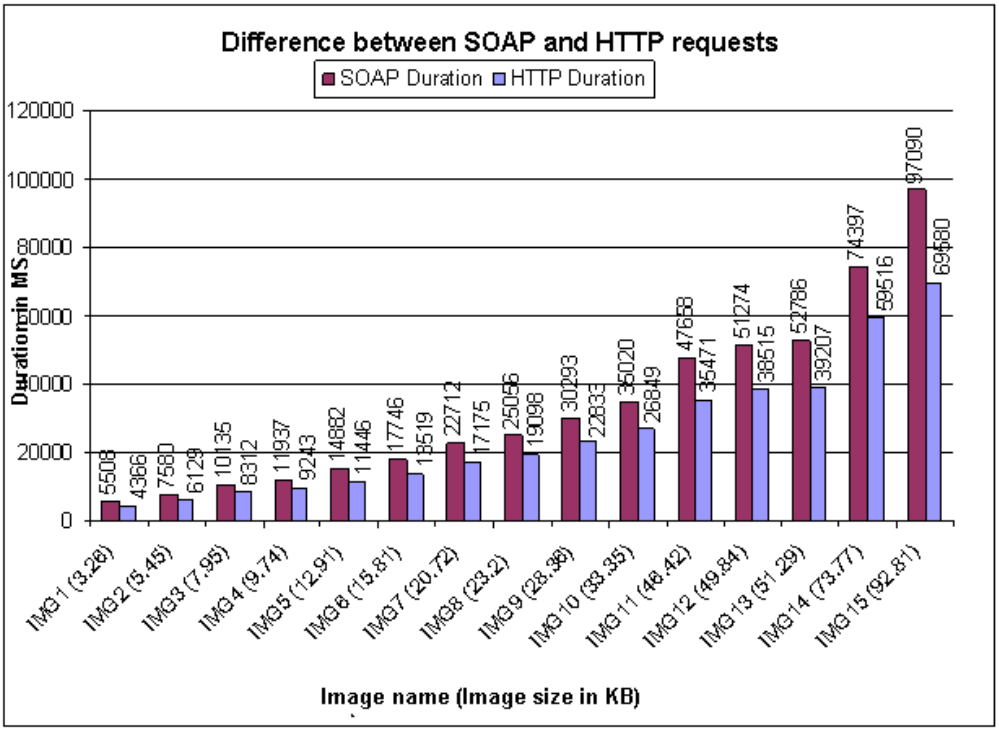


Figure 3.4: Difference between round-trip durations for the web service and HTTP requests (Adapted from [Srirama et al., 2006a])

for observing the behavior of the Mobile Host under concurrent requests from multiple clients, there by observing the robustness of the Mobile Host.

Apart from these services, other basic services like Echo service, ls service were also considered in the analysis. The ls service performs like the UNIX 'ls' command and helps in moving across the file system of the Mobile Host. The service was mainly used in browsing through the pictures of the Mobile Host, thus helping in Mobile Picture Service. The Echo service responds with the same message as the request message. The service has negligible $T_{process}$ time.

Performance Analysis of the Mobile Host

In the analysis, first the mobile picture service was used for observing the SOAP processing delay of the server. The results showed a significant difference (approximately 20%) between the time taken for web service access and the normal HTTP access. Figure 3.4 shows these times for the HSCSD connection.

The SOAP overhead and the Base64 encoding performed on the images before serialization of the response, has caused the size of the response to increase by more than 50%. The actual SOAP overhead caused to the size of the response is observed to be only

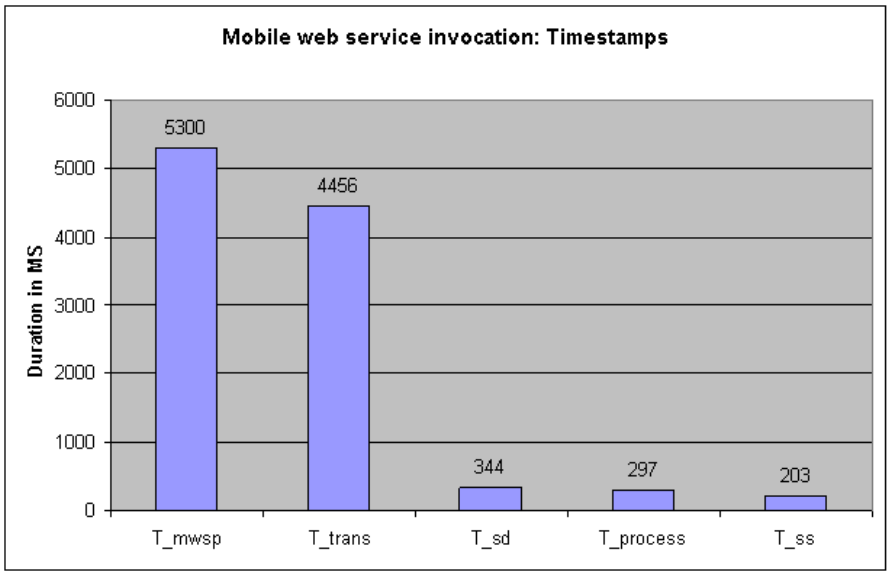


Figure 3.5: Time stamps for the GPS data provisioning service (Redrawn from [Srirama et al., 2006a])

578 bytes. This was observed using the Echo service with a single character request. The increase in the size has increased the transmission delay and there by increasing the delay in response.

In order to identify the actual times taken for different activities on the Mobile Host like T_{sd}, T_{ss}, T_{trans} etc., the location based service was requested by the client and the time stamps were taken at the Mobile Host while it was processing the request. These time stamps were later processed to get the operational time delays. Figure 3.5 shows the time delays of different activities, for the Location data provisioning service using the GPRS connection.

By observing the results from figure 3.5 and the detailed performance evaluation of the Mobile Host, we could conclude that service delivery as well as service administration can be performed with reasonable ergonomic quality by normal mobile phone users. 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. Similarly the comparison of results for HSCSD and GPRS connections suggested that the increase in transmission rates can increase the processing capability of the Mobile Host. This makes the performance of the Mobile Host directly proportional to achievable higher data transmission rates. Thus, the high data transmission rates achieved, in the order of few Mbps, through advanced mobile communication technologies in 2.5G, 3G and 4G, help in realizing the Mobile Host in the commercial environments [Srirama et al., 2006a].

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. Concurrent requests were generated for the services deployed on the Mobile Host, simulating multiple clients. The results of this regression analysis for checking the scalability of the Mobile Host are very encouraging and the Mobile Host was successful in handling up to 8 concurrent accesses for reasonable service like location data provisioning service with response size of approximately 2Kb. But the same analysis conducted for the Mobile Host with mobile picture service, where the response size is approximately 40 kb, could process only 3 requests. The main reason for not being able to process more mobile web service clients was due to the transmission delay. It was also observed that the concurrent access can affect the Mobile Host's ability to access internal and external resources. The comparison of the time stamps for the GPS data provisioning in unique and concurrent access revealed that the only drastic difference was at the $T_{process}$.

The study of the memory footprints revealed that memory usage was not a problem with Mobile Host, as most of the time, the amount of free memory was at least 20% of the total memory allocated for the JVM (Java Virtual Machine) (max value approximately 330 KB) and the "Out of Memory error" was never encountered during the execution of the tests. Approximately 200 data traces were observed as the experiments were repeated several times in order to have statistically valid results [Srirama, 2004].

3.1.4 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. The applications and usage scenarios of the Mobile Host are addressed in detail, throughout the coming chapters. Especially, chapter 7 on page 161 explains these applications with main focus at Mobile Host's usability analysis. This subsection briefly introduces some of the applications, to familiarize the reader with the scope of usage of the Mobile Host.

Several applications were developed and demonstrated with the Mobile Host, 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 [Srirama et al., 2006a]. The scope of the Mobile Host in m-learning (mobile learning) domain is also 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 [Chatti et al., 2006].

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 [Belov et al., 2005] 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. As already mentioned, the scenarios are explained in detail in section 7.1 on page 161.

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.

3.2 Similar Mobile Server Approaches

Web services are not the only means of providing services from devices like smart phones and PDAs. The provisioning can also be based on any distributed communication technology like Java Remote Method Invocation (RMI) or Jini, if the device supports the respective platform. Van Halteren et al have addressed nomadic mobile service provisioning, based on Jini technology. The Jini system architecture consists of three categories: *programming model*, *infrastructure*, and *services* [Sun Microsystems, 2001]. The infrastructure is the set of components that enables building a federated Jini system, while the services are the entities within the federation. The basic infrastructure consists of the discovery/join protocol and the lookup service. Discovery is the process by which a Jini-enabled device locates lookup services on the network and obtains references to them. Join is the process by which a device registers the services it offers with a lookup service. The programming model includes models for leasing, event notification, and transactions. The Jini infrastructure is built on top of the Java application environment. Figure 3.6 shows the architecture of nomadic mobile service provisioning.

The nomadic mobile service provisioning approach based on Jini proposes the Mobile Service Platform (MSP) as a supporting infrastructure which extends the SOA paradigm to

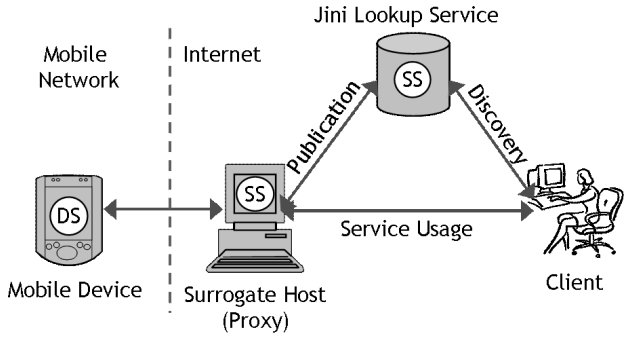


Figure 3.6: Proxy based NMS provisioning (Adapted from [Pawar et al., 2007])

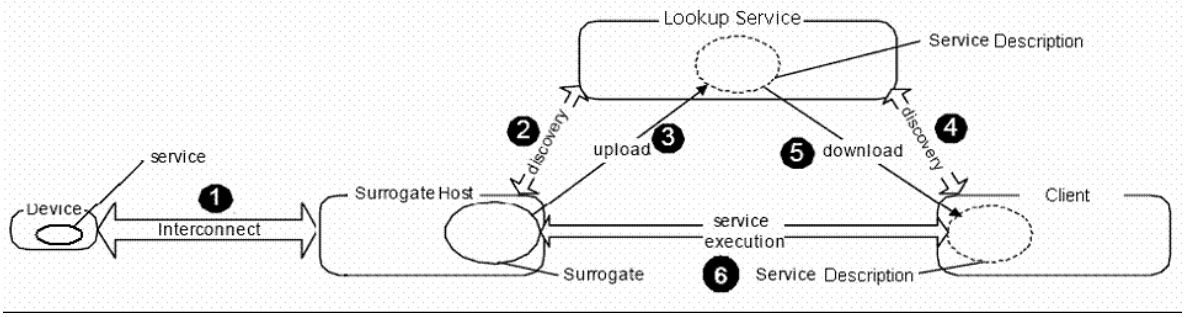


Figure 3.7: Elements of mobile service platform (Adapted from [van Halteren and Pawar, 2006])

the mobile device. MSP is a middleware that facilitates the development and deployment of innovative services on the mobile device for clients located anywhere in the Internet. The MSP design is based on the *Jini Surrogate Architecture Specification* which enables a device which can not directly participate in a Jini Network to join a Jini network with the aid of a third party [Sun Microsystems, 2007h]. MSP consists of an HTTPInterconnect protocol to meet the specifications of the Jini Surrogate Architecture and provides a custom set of APIs for building and running services on a mobile device. Using this architecture a service provided from the device is composed of two components: 1) A service running on the mobile device (referred to as a *device service*); and 2) A *surrogate service*, which is the representation of the device service in the fixed network. The surrogate (SS) functions as a proxy for the device service (DS) and is responsible for providing the service to the clients. The MSP supports the communication between the device service and the surrogate service. Thus using mobile service platform a service hosted on a mobile device can participate as a Jini service in the Jini network [van Halteren and Pawar, 2006]. Figure 3.7 shows the elements of the MSP and the stages during the lifecycle of NMS are numbered from 1 to 6.

For publishing a service in the network, the surrogate contacts the Jini lookup service for the service registration. After the lookup service is discovered either through unicast or

multicast discovery [Sun Microsystems, 2001], the NMS description is registered with the lookup service. The surrogate needs to periodically renew the NMS registration. In case the registration is not renewed for a certain time, the lookup service will discard it. For accessing the published service a client downloads a service proxy which communicates with the actual service, using any remote invocation protocol e.g. RMI. The surrogate often catches the response from the device service. However, to ensure that the client receives the latest data from the device service, some mechanism is necessary for the communication between the device service and surrogate, and regular synchronization of their respective data. For this purpose, the provisioning approach uses the HTTPInterconnect protocol, which defines the following three types of interactions between the device service and surrogate: 1) *One-Way* messaging allows for unacknowledged message delivery. 2) *Request-Response messaging* supports reliable message delivery. 3) *Streaming* interaction supports exchange of continuous data (streams). Each message has an operationID and sequenceID. Each operation offered by the service to its surrogate and vice versa, need to have a unique operationID, so each message can trigger a certain operation. The body of a message contains data specific to the operation to be performed by the message.

The approach has some advantages over mobile web service provisioning. Since the surrogate is located in the fixed network, it serves a potentially unlimited number of clients and thus minimizes the bandwidth usage in the mobile network. The clients are largely unaware of the fact that the environment in which the real service resides is resource constrained. However, splitting a service into a device service and surrogate also introduces a state synchronization problem. The surrogate must be aware of the change in the state of a device service. Most serious limitation of this approach is that, it is based on a proprietary protocol. The technology (Jini) is also fixed. So the client should be aware of Jini technology. Moreover the services are to be developed both for the surrogate and the device and changes are not propagated. The approach thus tightly fixes the service provided by the mobile device to platform (Java), protocol (HTTPInterconnect), technology (Jini) and surrogate host, thus seriously affecting the interoperability of the provided services. Then the only advantage left for the approach is the support for unlimited number of clients, which is easily achieved with proper QoS support for the mobile web service provisioning with the help of a mediation framework [Pawar et al., 2007]. The mediation framework is discussed in chapter 6.

3.3 Web Services Related Security Specifications

While service delivery and management from Mobile Host is technically feasible, as discussed from previous sections, the ability to provide proper Quality of Service (QoS), especially in terms of security for the Mobile Host is observed to be very critical. 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 over the network, 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 [Atkinson et al., 2002], Security Assertion Markup Language (SAML) [Cantor et al., 2005b] etc. exist, but not much has been explored and standardized in wireless environments. This section introduces some of the existing security standards and specifications in the web services domain. The details of the security analysis of mobile web services are addressed in section 4.1 on page 77.

- W3C is primarily responsible for XML Encryption [Eastlake et al., 2002a] and XML Signature [Eastlake et al., 2002b] standards.

- OASIS (Organization for the Advancement of Structured Information Standards) is an organization which has larger interest in web service specific standards and it owns WS-Security and SAML standards.

3.3.1 XML Encryption

XML encryption is the process of encrypting and decrypting digital XML content, using certain syntax and security algorithms. XML Encryption specification [Eastlake et al., 2002a] includes a standard syntax for representing the encrypted content within the XML, as well as information required to decrypt its contents at the receiving end. The content can be encrypted using the encryptions algorithms, also known as ciphers, like RCx ciphers from RSA Security [RSA Labs, 2007b], the *Data Encryption Standard (DES)* from National Institute of Standards and Technology (NIST) and other variants of DES [NIST, 1999]. Although *Secure Socket Layer (SSL)* [Freier et al., 1996] and *Transport Layer Security (TSL)* [Dierks and Allen, 1999] are the de facto standards for secure communication over the Internet, they are not suitable for multiparty interactions, a typical characteristic of XML and web service interactions. With XML encryption specification it is possible to encrypt only a part of the XML document like a single element or only the content of an element. It is also possible to encrypt different parts of the document with different keys, so that multiple recipients can decrypt only those portions that are intended for them. Thus XML encryption secures documents exchanged between web services.

The specification defines <EncryptedData> element as the core element that should enclose all encrypted XML data. EncryptedData becomes the root of the resulted document in case the whole document is encrypted. Its child element, <CipherData> encloses the actual cipher string that results from the encryption process. The encrypted data is placed in the <CipherValue> element. Alternatively, if the encrypted data is stored at a different server, <CipherReference> element contains the URI attribute that points to the location of the cipher data. The <EncryptedKey> element embeds relevant key information, required for decrypting the data, at the receiving end [Gokul, 2002].

3.3.2 XML Signature

XML signatures are digital signatures designed for use in XML transactions. The XML Signatures standard [Eastlake et al., 2002b] defines a schema for capturing the result of a digital signature operation applied to XML data. Signature of digital content is a two-stage process. In the first stage, the digital content is *digested* and the resulting value is placed in an XML element. In second stage, the digested value is picked and signed. Digesting the original content generates a small but unique encrypted value (digest) that takes less time to sign as compared to the original content. The content is digested using hashing algorithms like MD5 [Rivest, 1992] or SHA-1 [NIST, 1995]. MD5 (Message-Digest algorithm 5) after hashing produces a 128 bit message digest while SHA-1 (Secure Hash Algorithm) creates a 160 bit digest. XML signatures add authentication, data integrity, and support for non-repudiation to the data that they sign. Similar to XML encryption, XML signatures also allow signing only specific portions of the XML tree rather than the complete document. It is also possible to sign different parts of the document with different keys. They are relevant when a single XML document may have a long history and different parties or intermediaries work on different parts of the document each signing only those elements relevant to it, just like in web services.

After the XML data or part of the document is digitally signed, the resulting XML signature is represented by the <Signature> element. The <Signature> element refers to the signed element by using information in its <Reference> element. In addition to a reference the <Signature> element includes the method used to canonicalize the digital content and the algorithm used to generate the signature in the <SignatureMethod> child element. The <KeyInfo> element embeds relevant key information used to validate the signature [Verma, 2003].

3.3.3 WS-Security

The WS-Security specification [Atkinson et al., 2002] from OASIS is the core element in the 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. There are three types of security tokens, namely *Username/password*, *Binary*, and *XML tokens*, which can be attached to WS-Security header. The username token defines the way to pass the username and the optional password. If the token is not encrypted, the password should be transferred through secured channel. Binary tokens are required to convert binary data into text-encoded format. X.509 [Housley et al., 1999] and Kerberos [Neuman and Ts'o, 1994] certificates are examples of binary data. The default encoding format for these tokens is *Base64* encoding. XML tokens are the carriers of SAML assertions. Security tokens in combination with XML Encryption ensure confidentiality while security tokens in combination with XML Digital Signatures ensure integrity, of the SOAP messages. WS-Security standard is extensible. New security

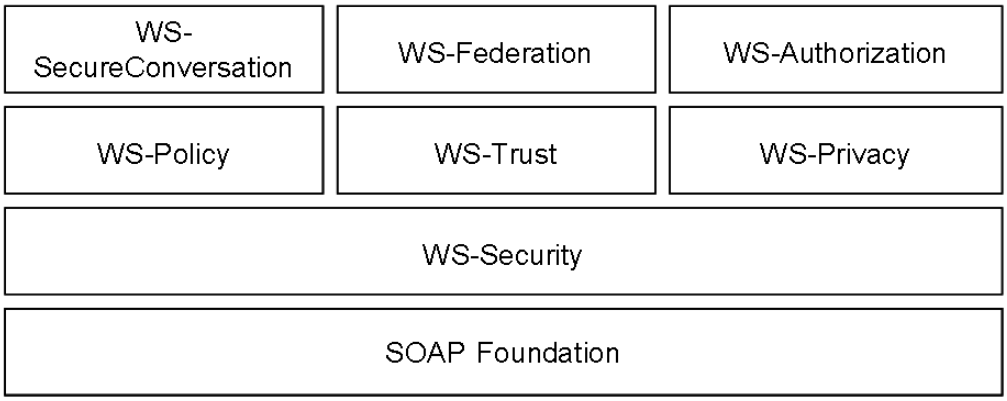


Figure 3.8: Web service security specifications (Redrawn from [IBM Corporation and Microsoft Corporation, 2002])

token and protocol types can be developed and added by defining additional profiles. These profiles will then make way to add the new tokens into the WS-Security framework.

Apart from WS-Security, web service security specifications also include *WS-Policy* which defines the rules for service interactions like service requestor and provider's requirements and capabilities concerning QoS of service interactions, such as security, transactionality, and communication reliability, *WS-Trust* which describes a framework for trust models that enables Web services to securely interoperate, by defining a set of interfaces that a secure token service may provide for the issuance, exchange, and validation of security tokens and *WS-Privacy* which states the maintenance of privacy of information like the organizational privacy practice statements. Specifications like *WS-SecureConversation*, *WS-Federation* and *WS-Authorization* are built on top of these set of basic specifications. WS-SecureConversation specifies how to establish and maintain secured session for exchanging data. It describes how to manage and authenticate message exchanges between parties, built on the concept of trust based on security tokens, including security context exchange and establishing and deriving session keys. WS-Federation defines rules of distributed identity and its maintenance. WS-Authorization describes how access policies for a web service are specified and managed. The set of web service security specifications are shown in figure 3.8 [IBM Corporation and Microsoft Corporation, 2002].

3.3.4 Security Assertion Markup Language (SAML)

Security Assertion Markup Language (SAML) is an extension of WS-Security from OASIS [Cantor et al., 2005b]. SAML specifies the language to exchange identity, attribute and authorization information between parties involved in web service communication in an interoperable way. SOAP over HTTP is used as the SAML request/response protocol transport mechanism. SAML requests and responses reside within the SOAP body.

SAML could help in achieving *single-sign-on (SSO)*. SSO is a mechanism where the authentication context of a consumer, can be maintained across multiple services. An example benefit of SSO is that visitors can use sites hosted by multiple companies, making it easier for online shoppers without having to login at each individual site.

SAML strives to provide a standard methodology to represent a principal's authentication and authorization information in XML format. The principal is a user or client application or a client web service requesting access to a specific resource. The basic components of interest in SAML are *assertions*, *protocols*, *bindings* and *profiles*. An SAML assertion is a declaration of certain facts (statements) about a principal. The third party authority that issues assertions is known as the *issuing authority*. Applications and services that trust the issuing authority and make use of its services, are called *relying parties*. SAML uses assertions to exchange security information like authentication information, across disparate systems and services. SAML request/response protocols tell how and what assertions can be requested i.e. they define XML schema and definitions for request/response model of transmitting security information. Bindings define the transportation of SAML protocols over SOAP over HTTP protocol [Cantor et al., 2005a]. As already discussed, the SAML request or SAML response will reside in SOAP body. Then SAML assertions are used along with the WS-Security element, which will reside in SOAP header. An SAML profile can be created using the bindings, protocols along with the assertion structure [Hughes et al., 2005].

The Extensible Access Control Markup Language (XACML) specification [Moses, 2005] provides the language to express access control mechanisms and policies within XML documents. It additionally standardizes mechanisms, algorithms and architectural components for managing these 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.

3.4 Web Service Related Scalability Specifications

Similar to providing proper security for mobile web services, providing proper scalability is also crucial in achieving suitable QoS from the Mobile Host. The layered model of web service communication, introduces lot of message overhead to the exchanged XML based SOAP messages. This consumes a lot of resources, since all this additional information is to be exchanged over the radio link. Moreover, XML-based messages are larger and require more processing than other protocols such as RMI, RMI/IIOP or CORBA/IIOP [OMG, 2004] and the binding requires more computation [Davis and

Parashar, 2002]. Hence only few requests can be handled by the Mobile Host. Thus for improving scalability of the Mobile Host the messages are to be compressed without seriously effecting the interoperability of the mobile web services.

The messages can be compressed with standard compression techniques like Gzip or XML-specific compression techniques like XMill to obtain smaller message sizes. Canonical XML [Boyer, 2001] standard targets the logical equivalence of these compressed XML messages. There is also an effort with the Fast Web Services [Sandoz et al., 2003], Fast Infoset standard draft [Sandoz et al., 2004], Efficient XML [AgileDelta, 2007], BinXML [Ericsson, 2003b] etc. to specify a binary format for XML data that is an efficient alternative to XML in resource constrained environments. Similarly, there is some effort with BiM (Binary Format for Metadata) [Heuer et al., 2002] standard for the binary encoding of MPEG-7 Metadata. This section introduces some of these XML data compression technologies. The details of the scalability analysis of the Mobile Host are addressed in chapter 4.2 on page 106.

3.4.1 XML Compression

XML has become a de facto standard for information transfer on the machine level and the application independent information storage. This is the reason why XML is used in web service communication to send data without having to worry about the incompatibilities between data representations in computer architectures, operating systems and services. But XML is quite verbose and is expensive, both in terms of storage space and processing time and resources required for parsing. Simple text compressors such as gzip could reduce the size, but they do not adapt to the specific demands of XML language such as duplication of names and tags that are spread throughout the document. GZip compression [loup Gailly, 2007] is based on numerous improvements to the LZ77 compression algorithm [Ziv and Lempel, 1977]. The basic idea behind LZ77 is to try to match current data in a dictionary of previously observed data. The dictionary and the current data are implemented using a fixed size window that slides over the data. When previously seen data are encountered, it is replaced by a reference to the dictionary, thus reducing the size of the text. There are mainly two drawbacks with the simple text compressors. Firstly, the redundant information in XML is spread over the whole document, but most text compressors work locally. For example, LZWM flushes its dictionary-based statistical table several times, when working through large files. While this is useful in text documents with few repetitions of words at the beginning and the end of the files, assuming this with XML is futile. Secondly the compressors lack the semantic understanding of different types of XML elements.

There are also some XML aware text compressors such as XMill [Liefke and Suciu, 1999], XMLPPM etc. XMill is a user-configurable XML compressor that groups text items with similar syntax and meaning and compresses them together. It separates structure, layout and data and distributes data elements into separate data streams (int, char, string, base64, etc). This distribution is user-definable. Simple text compressions like gzip,

bzip2 [Seward, 2005] compress these data streams. The grouping of data increases the redundancy of the document, and XMill can therefore, in many cases, achieve a higher degree of compression than for instance simple GZip compression. XMLPPM is a compressor for XML based on the Prediction by Partial Match (PPM) [Cleary and Teahan, 1997]. PPM predicts the upcoming character after certain strings, builds up a prediction table and compresses the document according to the probability of the contained strings. XMLPPM [Cheney, 2001] uses Multiplexed Hierarchical Modelling (MHM), i.e. several compressors (multiplexed) based on the XML structure and a hierarchical model of the document as context, rather than a sequence of characters. MHM is based on Encoded SAX (Simple API for XML) algorithm (ESAX). ESAX encodes the incoming SAX events into single bytes and stores the original names in a symbol table for later decoding. While these kinds of XML compressors achieve good results, they are generally quite slow.

Apart from these compression strategies there are also some attempts to optimize the XML thus reducing the size of the message [Rodriguez, 2002]. Some of these optimization principles include flattening the structure pattern, removing the additional comments, structuring node names and namespaces etc. These optimization mechanisms still preserve structurally equivalency of XML documents. Any two XML document are logically equivalent within an application context, if they only vary in physical representation based on syntactic changes permitted by XML and Namespaces in XML. Two structurally equivalent XML documents have a byte-for-byte identical canonical XML document. Canonicalizing an XML document requires only information that an XML processor is required to make available to an application.

3.4.2 Binary XML

Apart from XML compression, there exits several attempts at creating a standardized binary representation of XML data with equivalent power in terms of what can be expressed, but more efficient in terms of storage/transfer and parsing. The lack of a standard binary specification leaves the industry to invent own ad hoc standards. Fast Web Services is an initiative at Sun Microsystems aimed at the identification of performance problems in existing implementations of web services standards. It attempted at defining binary-based messages that consume less bandwidth and are faster and require less memory to be processed. Fast Web Services makes use of Fast Infoset documents for carrying the content of a SOAP message. The Fast Infoset standard draft specifies a binary format for XML infosets that is an efficient alternative to XML. An instance of this binary format is called a fast infoset document. Message Transmission Optimization Mechanism (MTOM) specification from W3C, describes an abstract feature for optimizing the transmission of a SOAP message by selectively binary encoding portions of the message, while still presenting an XML Infoset to the SOAP application [Gudgin et al., 2005]. Similar to the Fast Web Services, WAP Binary XML (WBXML) [Martin and Jano, 1999] is defined by the WAP Forum and is a part of the WAP 2.0 specification. The main purpose of WBXML is to create a compact representation of XML to reduce the

transmission size, with no loss of structure or semantics. WBXML does not include any support for compression of data. Millau [Girardot and Sundaresan, 2000] is a version of WBXML with support for compression.

The AgileDelta's Efficient XML provides a codec that encodes and decodes XML in a binary format that dramatically reduces the size. Similarly, BiM (Binary Format for Metadata) standard is defined for the binary encoding of MPEG-7 Metadata [Heuer et al., 2002]. MPEG-7 focuses on an XML based metadata system for describing the content of multimedia. There are two ways to transmit MPEG-7 information, either in *Textual Format (TeM)* or in a *binary form (BiM)*. Both provide similar functionality since both are able to transmit the description trees dynamically and incrementally. The benefit with BiM is as it compresses the XML code. BiM is designed in a way that it allows fast parsing and filtering of the XML data at the binary level itself, without having to decompress again. But, both decoder and encoder of BiM should be aware of XML schema in advance, to work properly.

[Ericsson and Levenshteyn, 2003] gives a comparison of different compression technologies for XML data (BiM is not considered in this analysis) and suggests BinXML as a good option to compress web service messages, considering compression ratio, processing time and resource usage. Based on this analysis, the study has adapted BinXML in the scalability analysis of the Mobile Host for compressing the mobile web service messages. BinXML is explained in detail in the next subsection. Even though there are many benefits with binary XML there is one serious limitation as the message looses its human readable self-descriptive nature of XML with the binary encoding and any existing standards and tools that rely on this mechanism will be seriously impacted.

3.4.3 BinXML

BinXML is a very simple binary representation of XML data [Ericsson, 2003b]. The encoding replaces each tag and attribute with a unique byte value and replaces each end tag with 0xFF. By using a state machine and 6 special byte values including 0xFF, any XML data with circa 245 tags can be represented in this format. The reserved codes are 0x00-0x03, 0xFE and 0xFF. The approach is specifically designed to target SOAP messages across radio links. BinXML uses a simple SAX parser and supports a variety of compression algorithms like LZO and GZip. BinXML has no support for schemas or DTDs and encoding/decoding is done based on the message itself. Each message contains enough information so that the receiving end can decompress it without prior knowledge of the message schema. The main version of BinXML is implemented in C. There is also an available Java port, which was adapted by the study for the J2ME version. The J2ME version uses the SAX parser provided by the Java ME Web services or JSR-172. Listing 3.1 on the following page shows a SOAP response message and its equivalent BinXML encoding is provided in table 3.1 on page 75.

The values, in the format <`0xYZ`>, shown in the table 3.1 on page 75 indicate the byte value of YZ in hexadecimal form. The size of the original SOAP message is 249 bytes

Listing 3.1: An example SOAP message

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV=
                    "http://www.w3.org/2001/12/soap-envelope/">
    <SOAP-ENV:Body>
        <expertRatingResponse>
            <status xsi:type="xsd:string">
                The rating was successfully received!
            </status>
        </expertRatingResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

excluding the white spaces (271 bytes with white spaces). The size of the encoded byte stream of the message is 195 bytes. This is a significant reduction (approximately 30%) to the size of the XML message. The compression ratio will be very significant where the SOAP message has repeated tags and very deep structure. Further analysis of BinXML is discussed while discussing the scalability of the Mobile Host in section 4.2.1 on page 109.

3.5 Mobile Web Services in P2P Networks

Once the Mobile Host was developed and its feasibility analyzed, as explained in section 3.1 on page 35, the study tried to find specific application domains, where Mobile Host can be adapted. The research mainly focused at mobile community support and pervasive applications. During this study, it was observed that most of the targeted collaborative applications, somehow converged to Peer to Peer (P2P) applications [Shirky et al., 2001], and P2P offered a large scope for many applications with Mobile Host. Not just the enhanced application scope, the P2P also offers several technical advantages to the Mobile Host like the improved mobile web service discovery mechanisms. This section introduces the P2P systems, their convergence with web services and the projects working at this convergence of the two technologies. The section also introduces Lucene, a tool used in advanced matching/filtering of mobile web services. The details of the P2P based mobile web service discovery analysis are addressed in section 5.3 on page 123.

3.5.1 P2P Technologies

P2P is a set of distributed computing model systems and applications used to perform a critical function in a decentralized manner. P2P networks are typically used for connecting nodes via largely ad-hoc connections. Peers are autonomous as they are not wholly controlled by each other or by some authority. Peers depend on each other for getting information, computing resources, forwarding requests. In P2P communications model each party has the same capabilities and either party can initiate a communication session.

Thus 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 enabling ad-hoc communication and collaboration. In the P2P world the peers collaborate through firewalls, NATs and proxies trying to connect to other peers.

P2P technology is being used in applications in the domains like entertainment systems, ubiquitous computing, pervasive computing and collaborative systems etc. P2P is also gaining popularity as low-cost individual computing technology. Three main classes of applications have emerged in the P2P environment over time. 1. *Content and file management P2P applications* like Napster [Carlsson and Gustavsson, 2001] and Gnutella [Ripeanu et al., 2002]. 2. *Parallelizable P2P applications* that split large tasks into smaller chunks that execute in parallel over autonomous peers like SETI@Home [Anderson et al., 2002] and Avaki [Grimshaw et al., 1997]. 3. *Collaborative P2P applications* that allow users to collaborate with each other without the help of central servers to collect and relay information, like Groove [Ozzie and Burton, 2003] and Skype [Skype, 2007]. Ad-hoc networks of mobile terminals are also participating in such P2P networks and applications, like in Magi [Bolcer et al., 2000]. [Milojicic et al., 2003]

P2P systems have evolved across time and provide a stable platform for many data and compute intensive applications. The applications can be categorized among three generations of P2P systems. The *first generation P2P systems* like Napster 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 providers using IP networks. But these centralized systems have single points of failure and produce giant communication traffic and storage on server resulting bottlenecks.

The drawbacks of first generation P2P systems lead to the *second generation of P2P systems*, also known as *pure P2P systems*, like Gnutella and Freenet [Clarke et al., 2002] which used a complete decentralized network. Unlike Napster, Gnutella would connect users directly to a group of other users and so on. The users on the system act as gateways to other users to find the data they need. For connecting to users, 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. Moreover the second generation P2P systems model suffered from major overhead generated by the binding messages and queries propagating around the Internet, leading to bottlenecks. For example in Gnutella the amount of broadcast messages transferred increases exponentially with a linear increase in the depth of the search [Kwok et al., 2003].

The *third generation P2P systems* like Skype [Skype, 2007], Kazaa [Kazaa, 2007], eDonkey [eDonkey, 2006], Bit Torrent [Pouwelse et al., 2005], etc. 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 and edge peers. Super peers have higher resource capabilities, when compared to edge peers, and act as relays for other edge peers and super peers. The super peers are organized dynamically and have

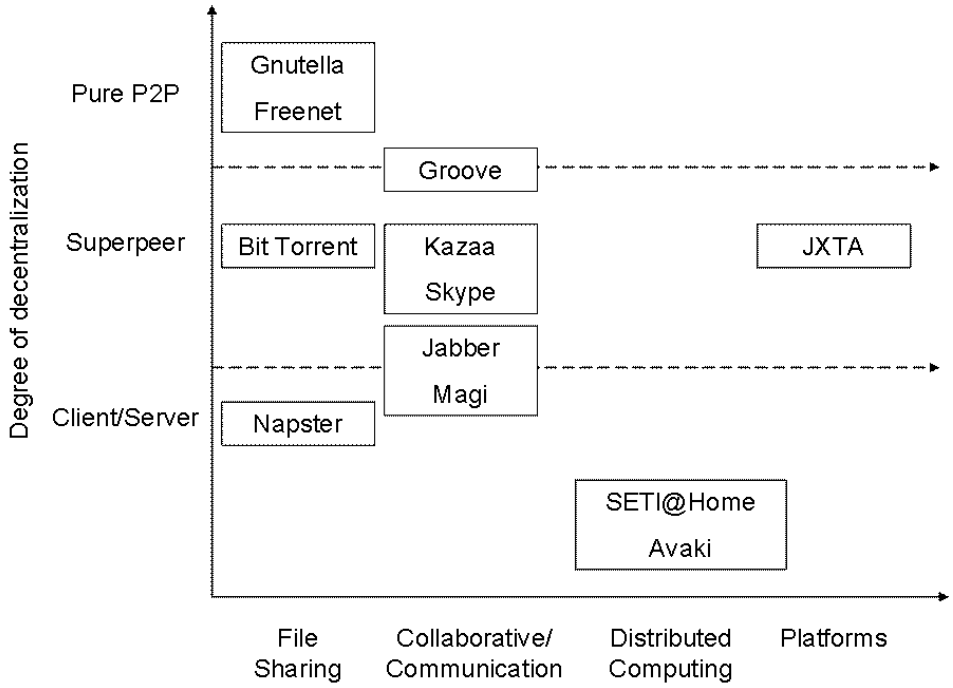


Figure 3.9: Classification of P2P technologies

abilities to traverse NAT and firewall. Only super peers participate in peer and resource discovery, which significantly decreases the stress caused to the network. Moreover several optimization methods are used for decreasing the message overhead [Harjula et al., 2004]. Figure 3.9 categorizes some of the well known P2P systems according to their characteristics and provided application types.

3.5.2 Convergence of Web Services and P2P

Analogous to web services, P2P systems can also leverage SOA and are also designed to enable loosely coupled systems. P2P services can also be described using XML schema and WSDL. The concept of services and the similarities of description stack of both P2P and web services make them comparable [Schneider, 2001]. Both the technologies have a heavy emphasis on distributed computing and target reliable delivery of services. For this, web services use centralized mechanisms to create highly available systems, such as clustered application servers with load balancing and hot swappable services, while P2P systems use distributed peers running the same service in a redundant manner with the first peer that hears about the request responding to the question. The major difference between the technologies being; web services are provided from well-known hosts, 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.

Apart from these major differences, in their pure form, the discovery mechanisms of both the technologies are also significantly different. In web services the discovery is based on centralized UDDI registries. P2P systems do not rely on centralized systems, and decentralize the information using advertisements. Peers perform discovery in a variety of ways including multicasts, inquiring about services that other peers know about, and using hubs, known as *rendezvous*, to act as a meeting place for peers with similar interests. Considering these benefits from P2P discovery and troubles with the centralized registries, the study developed a P2P based *mobile web service discovery* mechanism using the JXTA technology. The P2P discovery mechanism is explained in detail in section 5.3 on page 123.

For achieving this P2P discovery for mobile web services, the Mobile Host is to be adapted into the P2P network. For this many of the current P2P technologies like Gnutella, Napster and Magi are studied in detail. Most of these technologies are proprietary and are generally targeting specific applications. Only project JXTA offers a language agnostic and platform neutral system for P2P computing [JXTA Community, 2007b]. Next subsection briefly discusses the JXTA and introduces JXTA features required in publishing and discovery of mobile web services in P2P networks.

3.5.3 JXTA

JXTA technology, also known as *Juxtapose*, 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 technology is based on proven technologies and standards such as HTTP and TCP/IP, and is independent of programming language, networking platform, or system platform and can work with any combination of these. JXTA peers use XML as standard message format and create a virtual P2P network over the devices connected over different networks. Figure 3.10 on the following page shows the basic architecture of the JXTA. The *platform layer*, also called *JXTA core*, encapsulates minimal and essential primitives that are common to P2P networking. It incorporates building blocks that enable key mechanisms for P2P applications like discovery, transport with firewall handling, the creation of peers and peer groups, and associated security primitives. The *services layer* includes network services that are common in the P2P environment but may not be absolute mandatory for the operation of P2P network. For example searching and indexing, directory, storage systems, file sharing, distributed file systems, resource aggregation and renting, protocol translation, authentication, PKI (Public Key Infrastructure) services etc. The *applications layer* includes implementation of integrated applications, like P2P instant messaging, entertainment content management

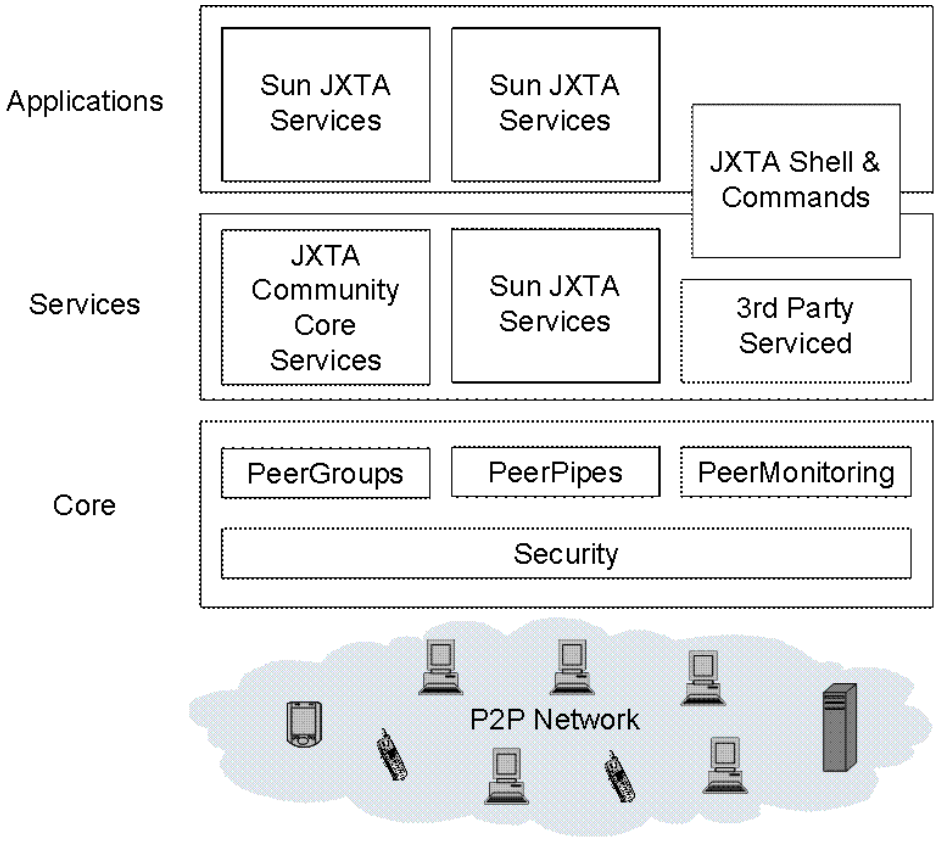


Figure 3.10: JXTA architecture (Redrawn from [Wilson, 2002])

and delivery, document and resource sharing, distributed auction systems, P2P Email systems etc. [Wilson, 2002]

Peers and Peer Groups

Any device like PC, PDA, smart phone etc., that implements one or more of the JXTA protocols could be a peer. 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 (Dynamic Host Configuration Protocol) based IP address. This peer ID will stay forever with that device even though the device supports multiple network interfaces like Ethernet, WiFi, and GPRS etc. for connecting to the P2P network. The peers can communicate with each other using the best of the many network interfaces supported by the devices. Moreover JXTA dynamically uses either TCP or HTTP protocols to traverse network barriers, like NATs and firewalls.

In JXTA peers, peer groups, pipes, contents, module classes, and module specifications are all represented with unique IDs. The IDs are represented in the form of URNs (Uniform Resource Names). URNs are a form of URI (Uniform Resource Identifier) that

are intended to serve as persistent, location- independent, resource identifiers. Like other forms of URI, JXTA IDs are presented as text. An example peer ID is shown below. It is a 32 byte unique ID. The first 16 bytes represent the peer group UUID (Universally Unique Identifier) to which the peer belongs. The following 16 bytes are the actual peer UUID. This allows finding out the group of a peer just by looking at the JXTA UUID Peer ID.

```
urn:jxta:uuid-596162616461626
14A78746150325033620EE00F29534A4585EADF4B7DCD5AC703
```

JXTA network supports different types of peers to be connected to the network. The general peers are called full-featured *edge peers*. A *minimal edge peer* can send and receive messages just like full-featured edge peer, but does not cache advertisements or route messages for other 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. Peers could self-organize into *peer groups*. The motivations to establish a peer group are e.g. to create a secure encoding and/or monitoring environment. The peers within a peer group agree upon a common set of services. Not all services within a peer group must be implemented by each peer. A peer just needs to implement services which are useful for him and use the implementation of the uncritical services provided by the default *Net Peer Group*. like Membership Service, Discovery Service, Pipe Service etc. Each peer in JXTA by default belongs to Net Peer Group.

Pipes

Pipes are virtual communication channels that are used to send messages by JXTA peers. They support the transfer of any object such as binary code, data strings, and Java based objects. Pipes are bound to specific endpoints, such as TCP port and associated IP address. The endpoints of a pipe are referred to as the *input pipe* and the *output pipe*. JXTA pipes offer two basic modes of communication, *point to point* and *propagate*. A point to point pipe connects exactly two pipe ends (an output pipe and an input pipe). The pipe is asynchronous and unidirectional. A propagate pipe sends messages from one output pipe to multiple input pipes. The broadcast is performed using IP multicast. IP Multicast is a one-to-many messaging protocol used to send one copy of data to a group address, reaching all recipients who are configured to receive it. Messages transferred across pipes are simple XML documents whose envelope contains routing, digest, and credential information.

Listing 3.2: An example peer group advertisement

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE jxta:PGA>
<jxta:PGA xmlns:jxta="http://jxta.org">
    <GID>
        urn:jxta:uuid-16A93B9296F44E5E8BF5EFA0CAEBC14B02
    </GID>
    <MSID>
        urn:jxta:uuid-DEADBEEFDEAFBABAFEEDBABE000000010306
    </MSID>
    <Name>
        WebServicesGroup
    </Name>
    <Desc>
        Created by Satish Narayana Srirama.
        The group advertises mobile web services to JXTA.
    </Desc>
</jxta:PGA>
```

Advertisements

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 other 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. Listing 3.2 shows an example JXTA advertisement, representing a peer group.

Modules

JXTA specifies *Modules* as a generic abstraction that allows peers to describe and instantiate any type of implementation of behavior 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 behavior. 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. JXTA modules are explained in detail in section 5.3 on page 123,

while discussing the mobile web service discovery in P2P networks.

Protocols

JXTA platform supports a set of basic protocols. Peers use these protocols to discover each other, advertise and discover network resources, and communication and route messages. The protocols are listed below and are being implemented in many different languages. Currently there are implementation in Java and C [Traversat et al., 2003].

- *Peer Discovery Protocol* enables peers to discover peer services on the network. The protocol can find peers, peer groups, and all other published advertisements.
- *Peer Resolver Protocol* allows peers to send and process generic requests. Queries can be directed to all peers in a peer group or to specific peers within the group.
- *Rendezvous Protocol* handles the details of propagating messages between peers. The rendezvous protocol is used by the Peer Resolver Protocol and the Pipe Binding Protocol to propagate messages.
- *Peer Membership Protocol* allows a peer to join or leave a peer group. The protocol supports the authentication and authorization of peers into peer groups.
- *Peer Information Protocol* provides peers with a way to obtain status information from other peers on the network.
- *Pipe Binding Protocol* provides a mechanism to bind a virtual communication channel to a peer endpoint.
- *Endpoint Routing Protocol* provides a set of messages used to enable message routing from a source peer to a destination peer.

JXTA for J2ME (JXME)

JXTA was initially targeting the standalone systems. Several requirements in the JXTA protocols like the caching and transfer of XML based advertisements and requirement of socket and datagram connections that are optional in J2ME, make it hard to implement JXTA on MIDP. The JXTA community has developed a light version of JXTA for mobile devices, called JXME (JXTA for J2ME) [Knudsen, 2002]. The goal of the JXME project is to bring JXTA functionality to 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 HTTP based binary communication with their proxies.

3.5.4 Projects Related to Convergence of Web Services and JXTA

Combining the web services and JXTA domains is not a completely new topic. Kevin Burton, during 2002, in the project *JXTA-SOAP* has implemented a transport mechanism for SOAP over JXTA. As discussed earlier, in section 2.4 on page 28, SOAP messages can be transmitted over any transport level protocol like HTTP, BEEP etc. JXTA-SOAP is a transport for Apache Axis, an open-source web services platform for Java. Axis has a plug-in architecture that allows supplementary transports than the default HTTP transport. JXTA-SOAP is such a transport plug-in to allow SOAP to work over JXTA. When JXTA-SOAP is used, the SOAP request/response messages are put in JXTA messages and sent through JXTA pipes, rather than put in HTTP messages and sent over TCP/IP. The project was previously referred to as the *JXTA Bridge* project. The JXTA-SOAP project is being maintained by Michele Amoretti of Distributed Systems Group (University of Parma, Italy) [JXTA Community, 2007a].

Similarly, [Hajamohideen, 2003] proposed a *Proxy Model* for the invocation of web services in JXTA network. The proxy model is shown in figure 3.11 on the facing page. The JXTA model specifies how services other than the core services can be published using the module advertisements and can be discovered using the discovery service. But the JXTA model does not mention how these services are invoked. The service description on how to invoke a service, the details of the operations and parameters are assumed by the client peer. The current JXTA model can use a service class implemented as local C++ or Java modules or as binary executables that invoke methods on the local class implementation. The proxy model adapts this feature of JXTA and uses the WSDL document to dynamically generate a proxy class. A WSDL compiler is used to generate these Java source files. Systinet WASP (Web Applications and Services Platform) [Allan, 2004] web services toolkit was used to invoke the remote web service using the proxy class. This proxy model seamlessly integrates the web service invocation into the JXTA model and creates an interface to access the web service using web service protocols in an implementation independent manner. The proxy class thus becomes a JXTA service that in turn can be advertised in the JXTA network. The JXTA client does not require prior knowledge of a web service or the invocation framework of any specific web services toolkit.

[Elenius, 2003] defined a mechanism for the service discovery in P2P networks. In this work, he showed that JXTA P2P networking infrastructure can be extended and integrated with the DAML-S (DAML Services) service ontology for service description, WSDL for interface descriptions, and SOAP for remote invocation. Subsequently, [Qu and Nejdl, 2004], with their *Edutella* product, discuss the exposing of existing JXTA services as web services; and also integrating web service enabled content providers into JXTA, using the proxy model. Edutella is a P2P semantic Web application, which is aimed to accommodate heterogeneous resource metadata repositories in a P2P manner and further facilitate the exchange of the resource metadata based on RDF (Resource Description Framework).

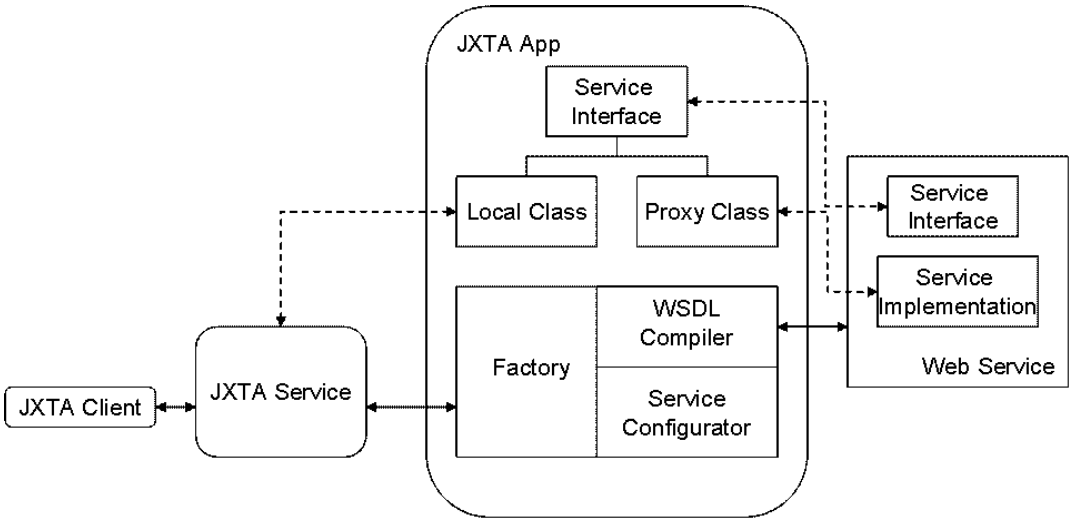


Figure 3.11: Proxy model architecture (Redrawn from [Hajamohideen, 2003])

Some similar approaches of integrating the web services, semantic web standards and JXTA P2P technology are adopted by *WS-Talk Project* Consortium [WS-Talk, 2007]. WS-Talk uses WSDL for web service description; DAML (DARPA (Defense Advanced Research Projects Agency) Agent Markup Language) which has now become OWL (Web Ontology Language) for ontology description; SOAP, for access and transfer; all based on JXTA using pipes or sockets as a transport layer. Figure 3.12 on the next page shows WS-Talk's mechanism of integrating web services into JXTA environment [Contreras et al., 2005].

3.5.5 Lucene

Lucene is the tool used in advanced matching/filtering of mobile web services [section 5.3.4] in the mobile web service discovery in P2P networks. It is an open source project hosted by Apache and provides a Java based high-performance, full-featured text search engine library [Cutting, 2007]. To search large amounts of text quickly, one must first index the text and convert it into a format that can be searched rapidly, eliminating the slow sequential scanning of each file for the given word or phrase. This conversion process is called *indexing*, and its output is called an *index*. Searching is the process of looking up words in an index to find documents where they appear. Lucene allows to add indexing and searching capabilities to user applications, and can index and make searchable any data that can be converted to a textual format. This means Lucene can be used to search and index information kept in JXTA advertisements, files, web pages on remote web servers, documents stored in local file systems, simple text files, Microsoft Word documents, HTML (HyperText Markup Language) or PDF (Adobe Portable Document Format) files, or any other format from which textual information can be extracted.

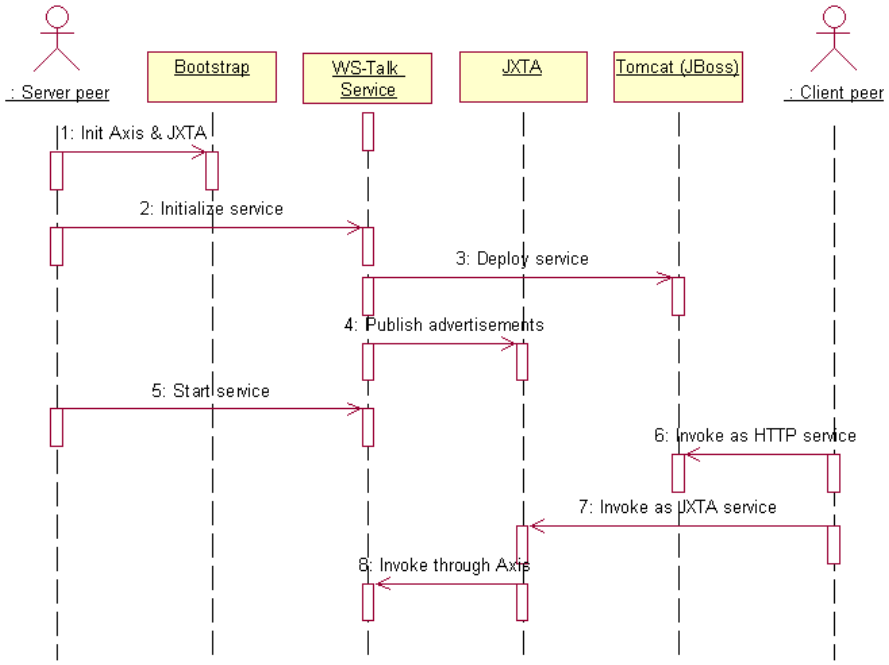


Figure 3.12: JXTA / Web services integration (Redrawn from [WS-Talk, 2007])

Java Lucene is a rapid and reliable tool and the product is being used by many well known websites like Wikipedia, an online encyclopedia, as well as in many Java applications. To build an Index Lucene uses different types of analyzers like StandardAnalyzer, WhitespaceAnalyzer, StopAnalyzer, SnowballAnalyzer etc. The analyzer breaks text fields up into index-able tokens and it is the core part of the Lucene. For example; StandardAnalyzer is a sophisticated general-purpose analyzer. WhitespaceAnalyzer is a very simple analyzer which just separates tokens using white space while StopAnalyzer removes common English words which are not usually useful for indexing [Gospodnetic and Hatcher, 2007].

3.6 Integrational Aspects of Mobile Web Service Provisioning

The Mobile Host's application, QoS and discovery research have identified the need for intermediary nodes helping in the integration of mobile web service provisioning deployment scenario. The deployment scenario is addressed in detail in section 6.2 on page 146. For realizing these intermediary nodes the study relied on the *Enterprise*

Service Bus (ESB) technology. This section explains the ESB technology, the Java Business Integration (JBI) specification and the open source ServiceMix tool used in realizing the integration framework.

3.6.1 Enterprise Service Bus

Enterprise networks commonly deploy disparate applications, platforms, and business processes that need to communicate or exchange data with each other. The applications, platforms and processes generally have non-compatible data formats and non-compatible communications protocols. This leads to serious integration troubles within the networks. The integration problem extends further if two or more of such enterprise networks have to communicate among themselves.

Enterprise Application Integration (EAI) was the first technology addressing these integration issues. EAI software follows the *hub and spoke model* and the software acts as a hub that translates data and messages between different applications. It uses adaptors that reformat incoming data to a canonical format that is understood by both EAI and outgoing adaptors. To implement the connection among the EAI components and to achieve internal integration, the technology used asynchronous message broker like JMS. EAI products were large, inflexible, hard to manage and were generally expensive vendor solutions (high cost, vendor lock-in) [IONA Technologies, 2005].

ESBs are the next developments in the enterprise integration and a standards-based ESB solves the integration problem without the drawbacks of the EAI solutions. ESB provides a set of infrastructure capabilities, implemented by the middleware technology, that enable the integration of services in an SOA [Keen et al., 2004; Chappell, 2004]. The enterprise service bus concept is not a product, but an architectural best practice for implementing a SOA. Gartner et al. defines enterprise service bus as a new architecture that exploits web services, messaging middleware, intelligent routing, and transformation. ESBs act as a lightweight, ubiquitous integration backbone through which software services and application components flow [Schulte, 2007].

ESB basically consists of a set of service containers that are interconnected with a reliable messaging bus. Generally in point-to-point integration solutions, each of the n components requires $n - 1$ interfaces for full communication among them, but with the bus solution each component requires only a single interface to the bus for global communication. The ESB offers dedicated infrastructure providing the capability to route and transport service requests to the correct service providers. The ESB supports multiple integration paradigms in order to fully support the variety of interaction patterns that are required in a comprehensive SOA between these service containers. So it has support for *service-oriented architectures* in which applications communicate through reusable services with well-defined and explicit interfaces, *message-driven architectures* in which applications send messages through the ESB to receiving applications and *event-driven architectures* in which applications generate and consume messages independently of one another. Figure 3.13 on the next page shows a high-level view of the ESB.

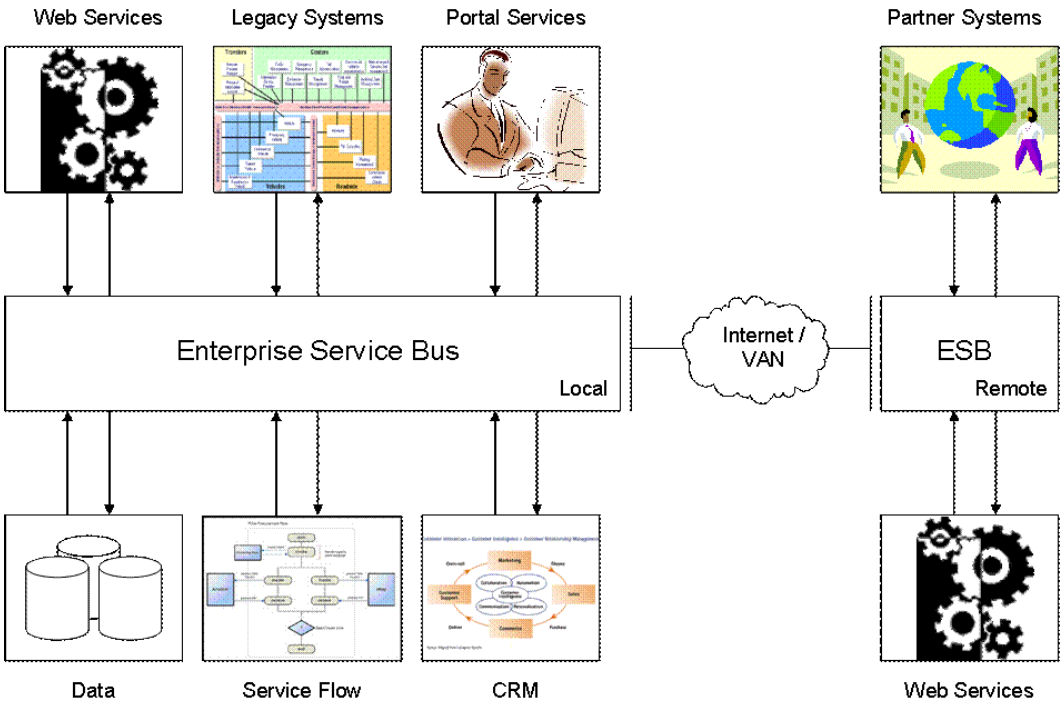


Figure 3.13: Enterprise service bus

ESB Features

The ESB provides a number of functionalities and features to achieve the task of integration [Thomas and Buckley, 2003; PolarLake, 2005]. These include:

- Transformation - The functionality provides the ability to map one data format onto another in order to ensure inter-operability between the various systems plugged into the ESB. The data formats are generally based on XML. The transformation can be performed by the ESB itself or delegated to external components. XSLT (Extensible Stylesheet Language Transformations) is very common approach for this sort of transformation. But XSLT engines are generally slow and hence ESBs offer more powerful transformation engines that handle the complex transformations needed for the application integration.

- Routing - A key prerequisite of any integration platform is to identify which data to process and where the data need to be sent. ESBs support content based routing and filtering, typically using XPATH (XML Path Language).

- Communication - The functionality supports the delivery of messages throughout the organization using different communication paradigms like *Synchronous/Asynchronous*, *Request/response*, *one-way*, *call-back* etc.

- Support for highly distributed environments, through which the ESB does not have to route all the messages through a central hub in order to apply routing and transformation.

- Security - The functionality supports secure transfer of messages among the participating service engines using mechanisms like user authentication, access controls etc.

- Orchestration - Orchestration is the process of automated coordination and management of composite applications components that participate in a business process. Generally ESBs achieve service composition, choreography and orchestration using BPEL (Business Process Execution Language) engines.

- Fault avoidance and fault tolerance using intelligent routing and exception handling. In ESBs a fault often not roll back all the work that has occurred at the time of a given business exception. Rather it will follow one or more rules and execute compensating transactions and/or accept the state of the process as satisfactory at the time of the exception.

- Transactionality - The functionality supports the reliable end-to-end delivery of messages among the ESB components.

- Support for pluggable services - These pluggable services can be provided by third parties and still interoperate reliably with the bus and the remaining components.

- Breadth of connectivity - The functionality supports the ability to connect to different type of systems like databases, enterprise applications e.g. SAP and legacy systems using well established standard mechanisms and tools.

ESB Products

Many ESB products exist like the Sonic Software, Artix, Cape Clear etc., which can be adapted for realizing the middleware framework. Of these tools, *Fiorano Software* and *Sonic Software* are two ESB products based on proprietary messaging middleware. Iona Technologies extends its legacy EAI architecture to achieve Iona Artix ESB. *PolarLake* and *FusionWare* take a server-centric, connector-based approach in their ESB product design. Only *Cape Clear* Software and *Cordys ESB* systems use a truly open and distributed SOA. [Borck, 2005] from InfoWorld provides a detailed survey of the products and their pros and cons with main focus at the achieved interoperability, management, scalability, security, supported features and their value for money. The survey recommends the Sonic SOA Suite among the contemporary ESB products. ESB products like *BEA AquaLogic Service Bus* [BEA AquaLogic, 2007] and *IBM WebSphere software* [IBM Corporation, 2007b] were not considered in this survey.

A recent analysis of the ESB products, provided at [MacVittie, 2006], suggests BEA AquaLogic Service Bus considering the features provided by each product, without considering any performance measurements. But most of these ESB products are based on proprietary message middleware or extensions to EAI architectures. SUN has defined JSR 208, *Java Business Integration (JBI)* specification, for enterprise application integration and ESB products like Service Mix and OpenESB [java.net, 2008] are based on this specification.

Observing these different ESB tools, standards and surveys, the thesis considered JBI and ServiceMix for the realization of *Mobile Web Services Mediation Framework (MWSMF)*. The details of the mediation framework are discussed in chapter 6 on page 145. The following subsections briefly introduce the JBI specification and the open source ServiceMix ESB tool.

3.6.2 Java Business Integration (JBI)

The Java Business Integration (JBI) specification (JSR 208) [Ten-Hove and Walker, 2005] defines a platform for building enterprise-class ESBs using a pluggable and service based design. The Java based standard allows building enterprise integration systems by using plug-in components which interoperate through mediated normalized message exchanges. Message normalization is the process of mapping context-specific data to a context-neutral abstraction to transport data in to a standard format. The normalized message exchange model is based on the web services description language (WSDL). The *normalized message* consists of the message content also called payload, message properties or metadata and optional message attachments referenced by the payload. The message attachments can be non XML data. JBI uses the normalized messages for interaction between consumers and providers. The basic architecture of JBI is shown in figure 3.14 on the next page. The *Normalized Message Router (NMR)* receives message exchanges from JBI components and routes them to the appropriate components for processing. This decouples the service producers from consumers. The NMR delivers messages with varying QoS, depending on application needs and the nature of the messages being delivered.

JBI Components

The JBI component framework provides a pluggable interface that allows components to interact with the JBI environment. The framework supports two types of components; *service engines* and *binding components*. Service engines are components responsible for implementing the business logic and they can be service providers/consumers. Service engine components generally provide support for content-based routing, orchestration, rules, data transformations etc. Binding components are used to send and receive messages across specific protocols and transports. The binding components marshall and unmarshall messages to and from protocol-specific data formats to normalized messages. Thus

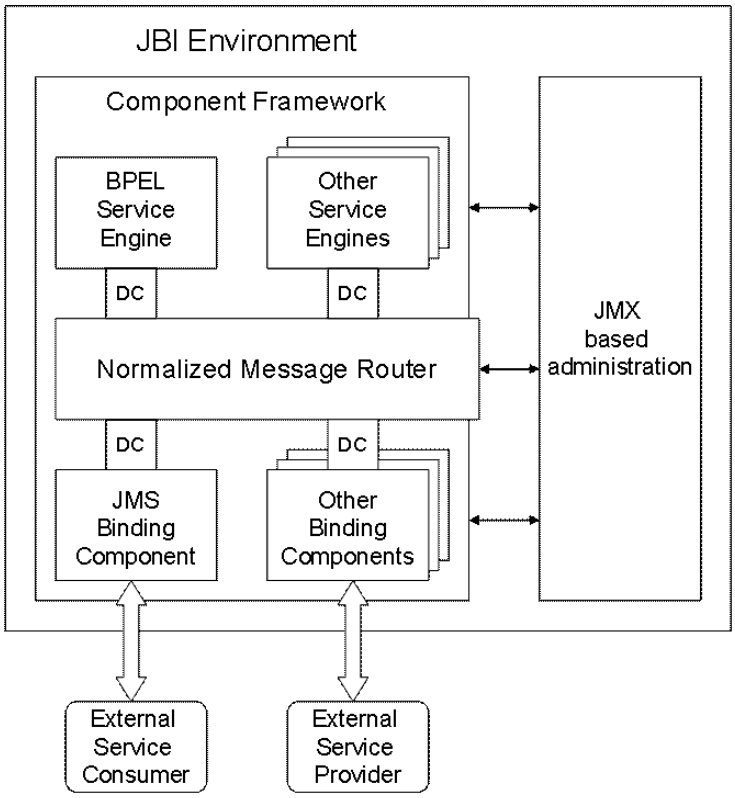


Figure 3.14: Java business integration architecture

binding components allow the JBI environment to process only normalized messages. In JBI component-specific artifacts like a single XSLT stylesheet or BPEL process deployed to a service engine or binding component are called *service units*. Service units are grouped into an aggregate deployment file called a *service assembly*. The service assembly includes a deployment descriptor that indicates the component into which each service unit is to be deployed. Binding components and service engines interact with the NMR via a *delivery channel*. A delivery channel is a bidirectional asynchronous communication pipe between a component and the NMR. Each component is provided with a single delivery channel, so the same channel can be used for both inbound and outbound communications. A service consumer uses its delivery channel to initiate a service invocation, while the provider uses its delivery channel to receive such invocations. A single component can act as both a service consumer and a service provider under different circumstances. The JBI environment provides deployment, control and monitoring features for the components through JMX (Java Management Extensions) based administration tools.

Endpoint activation is the process by which a service provider informs the NMR that it provides a service. After the endpoint activation NMR can route service invocations to that service. The activation process involves declaring a service endpoint with service QName plus endpoint name and providing a WSDL description of the activated endpoint.

In simple an endpoint refers to a specific address, accessible by a particular protocol, used to access a precise service. JBI supports two types of endpoints; *Internal endpoints* and *External endpoints*. Internal endpoints are exposed by service providers within the JBI environment accessible using the NMR APIs, while external endpoints are endpoints outside the JBI environment that are exposed by binding components. The binding components act as service consumers and expose an internal endpoint for the use of external service consumers. In JBI, endpoints can be referred *implicitly* where the NMR selects the endpoint based on the required service type or *explicitly* where a consumer component chooses the endpoint based on its own logic and configuration or *dynamically* where an endpoint reference is used within a message exchange to provide a call-back address that the service provider should use to send further message exchanges [Apache ServiceMix, 2007; Vinoski, 2005].

JBI Message Exchange Patterns

Service invocation in JBI refers to an instance of an end-to-end interaction between a service consumer and a service provider, involving the swapping of *message exchanges* between the components and the NMR. The message exchange is the container for the normalized message and the state of the service invocation. A consumer component initiates the exchanges by creating a new message exchange instance. Each operation that a service provider makes available has a particular *message exchange pattern (MEP)* associated with it.

JBI supports four types of MEPs.
1.) *In-Only MEP* describes a one-way messaging pattern. In this the consumer component initiates the request by creating a new message exchange instance and sending it to the NMR. The NMR queues the message exchange instance for delivery to the respective provider. The provider component accepts the instance from the NMR, processes the request message in the message exchange, and completes the MEP by setting the status of the message exchange instance to *done*, and sending the instance to the NMR. The NMR routes the instance back to the consumer component, which accepts the notification.
2.) *Robust In-Only MEP* is almost the same as In-Only MEP, with one extension. In this MEP, the provider may respond with a fault if it fails to process the request. Here the done status for the message exchange is assigned by the consumer.
3.) *In-Out MEP*: The consumer issues the request to provider, with the expectation of a response. The Provider may respond with a valid response or with a fault if it fails to process the request. In any case the done status for the message exchange is assigned by the consumer component.
4.) *In-Optional-Out MEP*: The consumer issues a request to provider, which may result in a response (The provider may set done status without responding to the consumer). This is an extension of In-Out MEP with both consumer and provider having the option of generating a fault in response to a message received during the interaction [Ten-Hove, 2006].

3.6.3 ServiceMix

ServiceMix is an ESB based on JBI specification and it combines the functionalities of both the SOA and the Event Driven Architecture (EDA) to achieve an agile, enterprise ESB. Released under the Apache license, ServiceMix is an open source ESB and SOA toolkit built on JBI semantics and API [Apache Software Foundation, 2007c]. The open source and open standards-based features of ServiceMix allow for low entry cost, maximum flexibility, reuse, and investment protection. The tool supports any number of third party vendor supplied components and protocol bindings that conform to the JBI open standard specification. These components and bindings not only interoperate among each other and external applications via the ServiceMix ESB, but can easily be replaced with alternate components that provide the same or enhanced services, without affecting the final deployment scenarios of the applications.

As mentioned already, ServiceMix allows services to operate in an event driven technique as well i.e. the services are decoupled and the providers listen for service requests on the bus. ServiceMix supports event driven architecture for events occurring both internal and external to the bus. In other words, JMS binding components can listen for the arrival of messages, i.e., the *event* on topics or queues which are external to the bus, while other components in the ESB can listen for these messages on the normalized message bus itself. The bus is also responsible for quality of service (QoS) features such as message persistence, guaranteed delivery, failure handling, and transaction support.

ServiceMix is *lightweight* and easily embeddable in the enterprise network. Lightweight design patterns let you loosen coupling between objects and integrate services without forcing code into business logic or the domain model [Tate, 2005]. ServiceMix is lightweight with integrated Spring support [Johnson, 2005] and it can be run at the edge of the network (inside a client or server), as a standalone ESB provider or as a service within another ESB. ServiceMix can also be used in a Java Standard Edition or in a Java Enterprise Edition application server. ServiceMix is completely integrated with JBoss [JBoss, 2007] and Apache Geronimo [Apache Geronimo, 2007] and lets the applications deploy JBI components and services directly into Geronimo. Some of the service components included with ServiceMix comprise rules-based routing via the Drools rule engine, Business Process Execution Language (BPEL) support for Web Services BPEL via PXE (preboot execution environment), timer integration via the Quartz library, transformation using Extensible Stylesheet Language Transformations (XSLT), a client API for working with JBI components and services and etc. [Apache ServiceMix, 2007; Hanson, 2005]

3.7 Summary

This chapter introduced the mobile web service provisioning concept and explained the details of the developed Mobile Host with its thorough performance and application analysis. The chapter later discussed the security and scalability related standards and

specifications for mobile web services. The chapter also introduced the concept of mobile web services in P2P networks and provided the basics of P2P technology and JXTA platform. Later sections of the chapter discussed the convergence of web services and P2P technologies with related state of the art projects. The chapter later discussed the integration issues for mobile web services and introduced the ESB technology. The JBI specification and open source ServiceMix tool used in realizing the integration framework are also explained in detail.

<0x01>	New tag
<0x10>	The new tag is assigned code 0x10
SOAP-ENV:Envelope	Name of the new tag
<0x00>	End of string
<0x03>	New attribute
<0x10>	Code of the new attribute (attribute code space), so 0x10 is free
xmlns:SOAP-ENV<0x00>	Name of attribute + End of string
http://www.w3.org/2001/12/ soap-envelope/<0x00>	Value of the attribute + End of string
<0xFE>	Soft end of tag. Represents the > in XML
<0x01>	New tag
<0x11>	Code 0x11
SOAP-ENV:Body<0x00>	Name of the new tag + End of string
<0xFE>	Soft end of tag
<0x01>	New tag
<0x12>	Code 0x12
expertRatingResponse<0x00>	Name of the new tag + End of string
<0xFE>	Soft end of tag
<0x01>	New tag
<0x13>	Code 0x13
status<0x00>	Name of the new tag + End of string
<0x03>	New attribute
<0x11>	Code of the new attribute 0x11
xsi:type<0x00>	Name of attribute + End of string
xsd:string<0x00>	Value of the attribute + End of string
<0xFE>	Soft end of tag.
<0x02>	String
The rating was successfully received!<0x00>	Value of string + end of string
<0xFF>	Hard end of tag (/> in XML)
<0xFF>	Hard end of tag
<0xFF>	Hard end of tag
<0xFF>	Hard end of tag

Table 3.1: BinXML encoding of the SOAP message shown in listing 3.1 on page 56

4 Mobile Host: QoS Extensions

From chapter 3 we can derive that service delivery and management from Mobile Host is technically feasible. But 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. In terms of scalability the Mobile Host has to provide alternative means for the verbose XML based web service communication in wireless domain. This chapter addresses the work performed in the mobile web services QoS domain.

4.1 Security Analysis of the Mobile Host

Mobile Host, as a provider of web services, has to offer secure and reliable communication in the vulnerable and volatile mobile ad-hoc topologies. Moreover with the easily readable mobile web services over the network, 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 [section 3.3.3], SAML [section 3.3.4] 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. In the security analysis of the Mobile Host, the thesis tried to realize some of the existing security standards in the mobile web services domain.

4.1.1 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. As discussed already, 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 security for the web service communication. Moreover, many legitimate intermediaries might exist in the web service communication making the security context requirement to be from end-to-end. For example a purchase order from mobile user might go through different processing subsystems like shipping, billing etc., which might add some specific Ids and details to the message headers. Hence the traditional point-to-point security technologies like

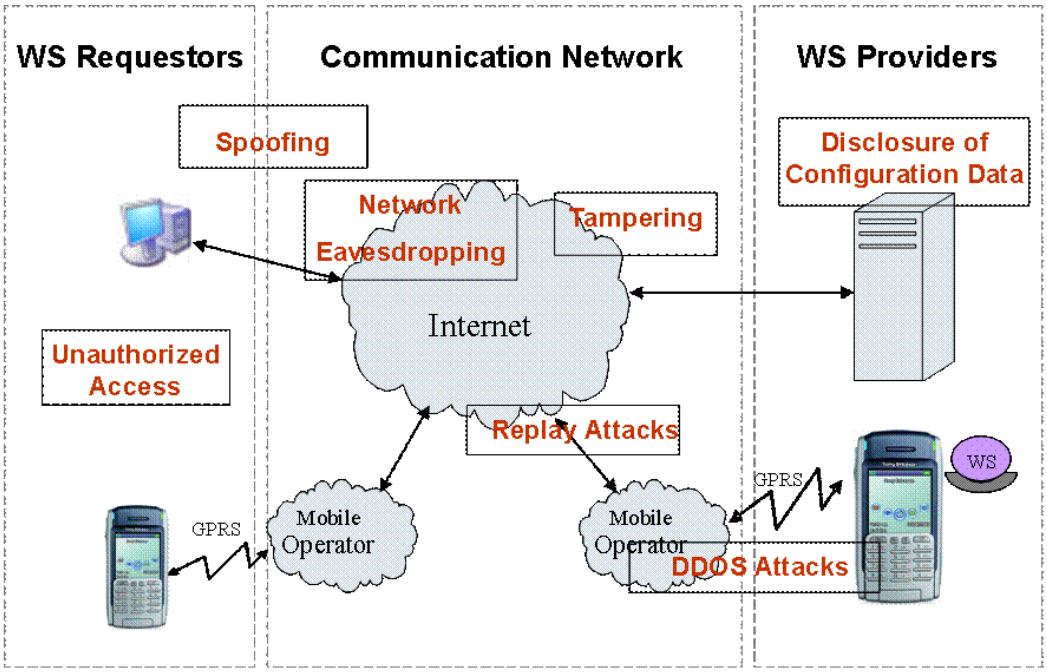


Figure 4.1: Typical security breaches in mobile web services (Adapted from [Srirama et al., 2007a])

the SSL [Freier et al., 1996], HTTPS (Hypertext Transfer Protocol over Secure Socket Layer) [Rescorla, 2000] and full encryption provided by the 3G technologies like UMTS communication technology [Keon et al., 2007] can't be adapted for the mobile web services domain. These security 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.

Figure 4.1 depicts some of the typical security breaches in web service and wireless environments, across the mobile web services domain [Meier et al., 2003]. *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 4.1 on the preceding page, is the *disclosure of configuration data*. Generally, WSDL documents reveal a 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 4.2 on the following page. Secured message transmission is achieved by ensuring *confidentiality* and *data integrity*, while *authentication* and *authorization* 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.

4.1.2 Security Analysis Design Model

To secure the communication of mobile web services provisioning, first the study analyzed the adaptability of WS-Security in the mobile web services domain [Srirama et al., 2007a, 2006b]. 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. Similar performance evaluation of WS-Security for PCs is provided at [Liu et al., 2005; Tang et al., 2006]. Especially Liu et al conducted detailed performance tests for different operations (Signing vs. Verifying and Encryption vs. Decryption) and different algorithms (MD5 vs. SHA1 and AES (Advanced Encryption Standard) vs. DES (Data Encryption Standard) etc.).

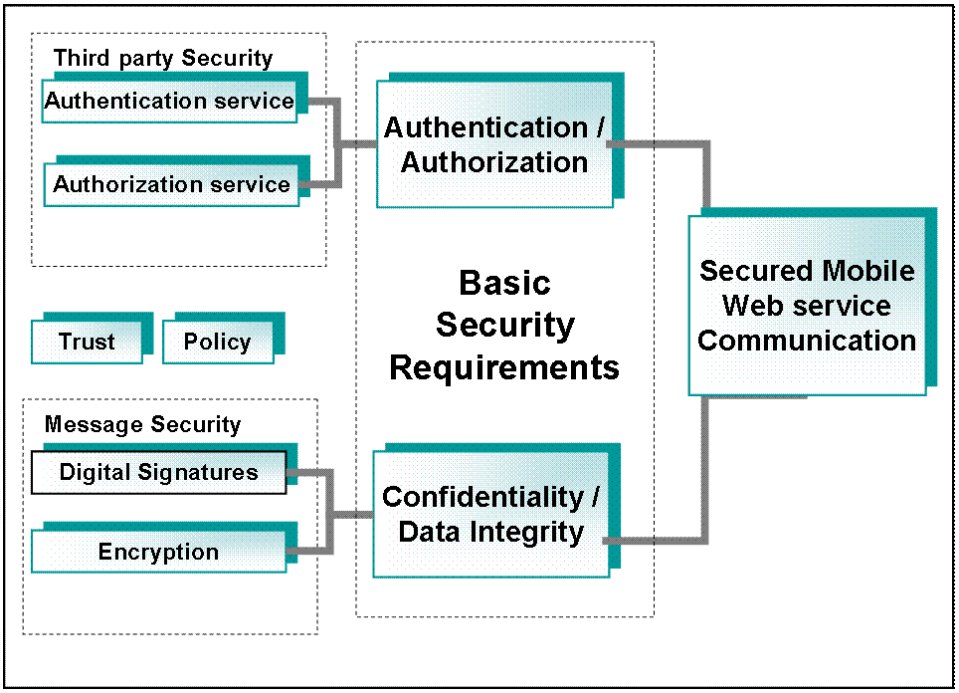


Figure 4.2: Basic security requirements for mobile web services (Adapted from [Srirama et al., 2007a])

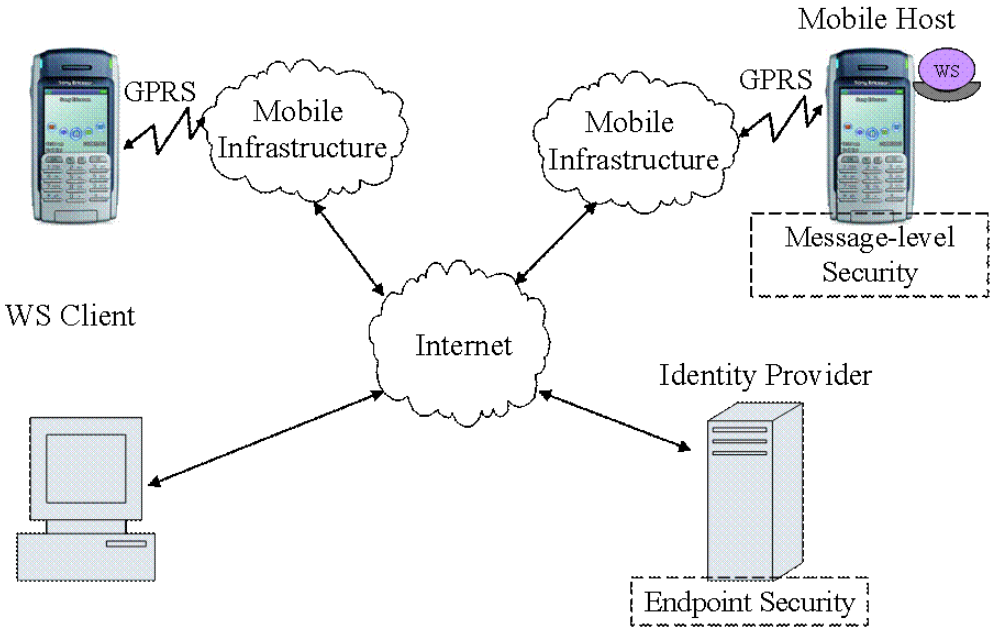


Figure 4.3: Proposed security realization scenario of Mobile Host (Redrawn from [Srirama et al., 2007a])

Figure 4.3 on the preceding page depicts the architecture considered by the study to analyze the basic security principles for the Mobile Host. The Mobile Host was developed and deployed on a smart phone. Once a web service is deployed with the Mobile Host; any web service client (standalone or mobile) can invoke the service. In the invocation process, 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 then 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. Later 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. The mechanisms were realized for the PersonalJava based Mobile Host using some of the features provided by the platform like the TaskSwitch [Bloor, 2002]. 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. The study considered password based over PKI based authentication, due to platform and resource limitations. 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 for storage. An alternative for this scenario is studied, where the Mobile Host generates an authentication request to a standalone web service, deployed on an Axis [Apache Software Foundation, 2007d] 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. Realization of single sign on, where identity and credentials can be maintained for multiple sessions and parties is also studied and is addressed in section 4.1.8 on page 94. The study also tried to automate the authentication and authorization processes by using Semantic Based Access Control (SBAC) mechanism [Srirama and Naumenko, 2007]. The study is introduced in section 4.1.9 on page 96.

The following subsections provide the 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

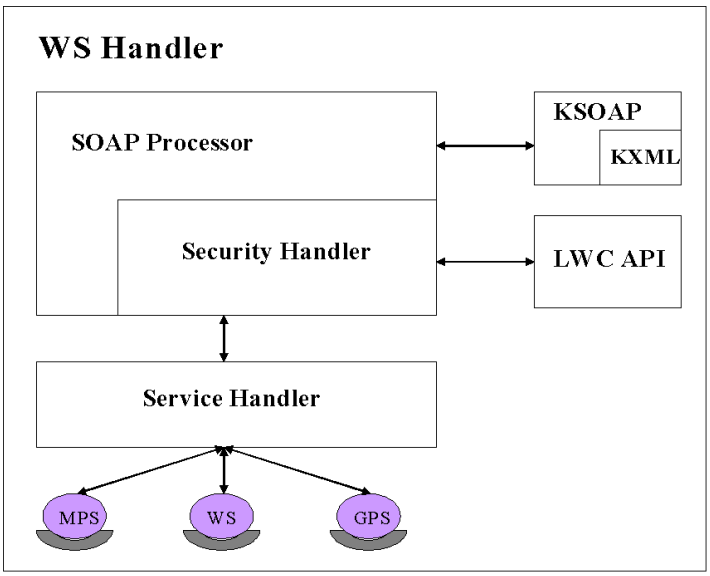


Figure 4.4: Web Service Handler of the Mobile Host (Adapted from [Srirama et al., 2007a])

scenario for securing mobile web services communication. This has left 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

4.1.3 Security Analysis Implementation Model

To analyze the WS-Security for Mobile Host, the study used two Sony Ericsson P910i smart phones as web service requestor and the Mobile Host. The smart phones had an internal memory of 64 Mb and ARM9 processor clocked at 156MHz. The phones were connected to the Internet using a GPRS connection. The PersonalJava based Mobile Host was rebuilt using J2ME, and the J2ME based Mobile Host was considered for the security analysis. The P910i device supports MIDP2.0 with CLDC1.0 configuration. For cryptographic algorithms and digital signers, Java based light weight cryptographic API from *Bouncy Castle crypto package* [Bouncy Castle, 2007] was used. KSOAP2 [section 2.3.2 on page 22] was adapted according to the WS-Security standard and was utilized to create the request/response web service messages.

The web service security enabled *WS Handler* component of the Mobile Host is shown in figure 4.4 on the preceding page. The WS Handler receives the web service messages from the HTTP interface of the Mobile Host, as explained in section 3.1.1 on page 36. 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 component manages the full message-level security and assists in end-point security.

To analyze *confidentiality* of the mobile web service message, the message was ciphered with *symmetric encryption algorithm* and the generated symmetric key is exchanged by means of *asymmetric key exchange method*. The message was tested against various symmetric encryption algorithms [RSA Labs, 2007b] including the WS-Security mandatory algorithms, namely, TRIPLEDES [RSA Labs, 2007d], AES-128, AES-192 and AES-256 [RSA Labs, 2007a]. The PKI algorithm used for key exchange was RSA-V1.5 [Rivest et al., 1978] with 1024 and 2048 bit keys. Upon successful analysis of confidentiality, the study tried to ensure data integrity of the messages. The messages were digitally signed and were evaluated against two signature algorithms, DSAwithSHA1 (DSS) [RSA Labs, 2007c] and RSAwithSHA1 with 1024 and 2048 bit keys. Later, the effect of signing on top of encryption was also studied, considering the best algorithms from the individual analysis of encryption and signing. Note that, as already mentioned, all the algorithms mentioned above have been implemented using the Java based light weight bouncy castle cryptographic API, hence the performance results are directly dependent on the API.

The study considered the location data provisioning service explained in section 3.1.2, as the test case service for the analysis. All of the test cases for message level security 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. The sizes in the range of 1-10Kb are considered as most of the relevant services considered in the study like the location information service have message sizes in the range < 3Kb. The different response sizes were considered as the size of the message was the only relevant parameter that would affect the performance of the Mobile Host handling secured mobile web service messages. Response sizes more than 10 Kb were not considered as, a service of this much message size, which requires encryption and signing of the messages could not be envisioned in the mobile phone environments, other than the digital rights protection of the pictures being exchanged. The web service request contained the <responseSize> element, which specifies the response size of the message being expected in Kb. The response was appended with an extra element <bodyPadding> to fill the remaining size of the response. The typical request and response of the test case mobile web service are shown in listing 4.1. 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.

Listing 4.1: Request and response messages of the location data provisioning service

```
<soapenv:Envelope xmlns:soapenv="..." xmlns:xsd="..." xmlns:xsi="...">
<soapenv:Body>
  <GPSProvider soapenv:encodingStyle
                  ="...">
   <responseSize xsi:type="xsd:int">
       1 </responseSize>
  </GPSProvider>
 </soapenv:Body>
</soapenv:Envelope>/


<SOAP-ENV:Envelope ...>
 <SOAP-ENV:Body ...>
<GPSProvider xmlns="ssn:SSNServer"
     id="o0" SOAP-ENC:root="1">
 <result>
  <Longtitude xsi:type="xsd:int">
     606428</Longtitude>
  <Latitude ...>5079068</Latitude>
  <Altitude ...>22</Altitude>
  <Speed ...>444</Speed>
     <Status ...>1</Status>
  <Comment xsi:type="xsd:string">
     </Comment>
 </result>
 <Request-ID ...>2</Request-ID>
 <bodyPadding xsi:type="xsd:string">
     ...</bodyPadding>
</GPSProvider>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.1.4 Security Analysis Performance Model

To analyze the performance of the Mobile Host with the security load, the performance model of Mobile Host, explained in section 3.1.3, was extended to embrace the security intervals. The durations of different activities across the mobile web service invocation cycle are observed. Figure 4.5 on page 86 shows the operations and respective time stamps in secured mobile web service invocation cycle. 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. With the added security parameters, the total time taken for this mobile web service invocation (T_{mwsp}) constitutes, the time taken by client for constructing valid SOAP message (T_{cc}), the time taken to encrypt the message with security information according to WS-Security standard (T_{reqec}), the time taken to serialize the encrypted message (T_{reqs}), the time taken to transmit the SOAP request to Mobile Host

(T_{reqt}), the time taken for de-serializing the XML based SOAP request message at the Mobile Host (T_{reqd}), the time taken to decrypt the request message (T_{reqdc}), the time taken by the Mobile Host to execute the respective business logic and to populate the response ($T_{process}$), the time taken to encrypt the response message with security information (T_{resec}), the time taken for serializing the encrypted response message back to XML data streams (T_{ress}), the time taken to transmit the SOAP response back to the client (T_{rest}), the time taken to de-serialize the response at the client (T_{resd}), the time taken by the client to decrypt the response message (T_{resdc}), and lastly the time taken by the client to process the response (T_{cp}). The invocation process is shown in figure 4.5 on the following page and the total time taken for the mobile web service invocation is given in equation 4.1.

$$T_{mwsp} = T_{cc} + T_{reqec} + T_{reqs} + T_{reqt} + T_{reqd} + T_{reqdc} + T_{process} + T_{resec} + T_{ress} + T_{rest} + T_{resd} + T_{resdc} + T_{cp} \tag{4.1}$$

The exact estimation of the T_{reqt} and T_{rest} time is not possible as their calculation process needs the synchronization of time stamps of both Mobile Host and mobile web service client. Moreover these transmission times were observed while analysing the performance of the Mobile Host [section 3.1.3] and those results showed that they constitute 90% of total invocation cycle. 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.

4.1.5 Evaluation of the Security for Mobile Host

The main idea of this 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 (International Data Encryption Algorithm) [Lai, 1992] with 128 and 256 bit keys, DES [NIST, 1999] 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 4.6 on page 87 summaries the results of the encryption analysis and shows the comparison of latencies for different encryption algorithms with keys exchanged using RSA 1024.

The results suggest that AES 192 encryption turns out to 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

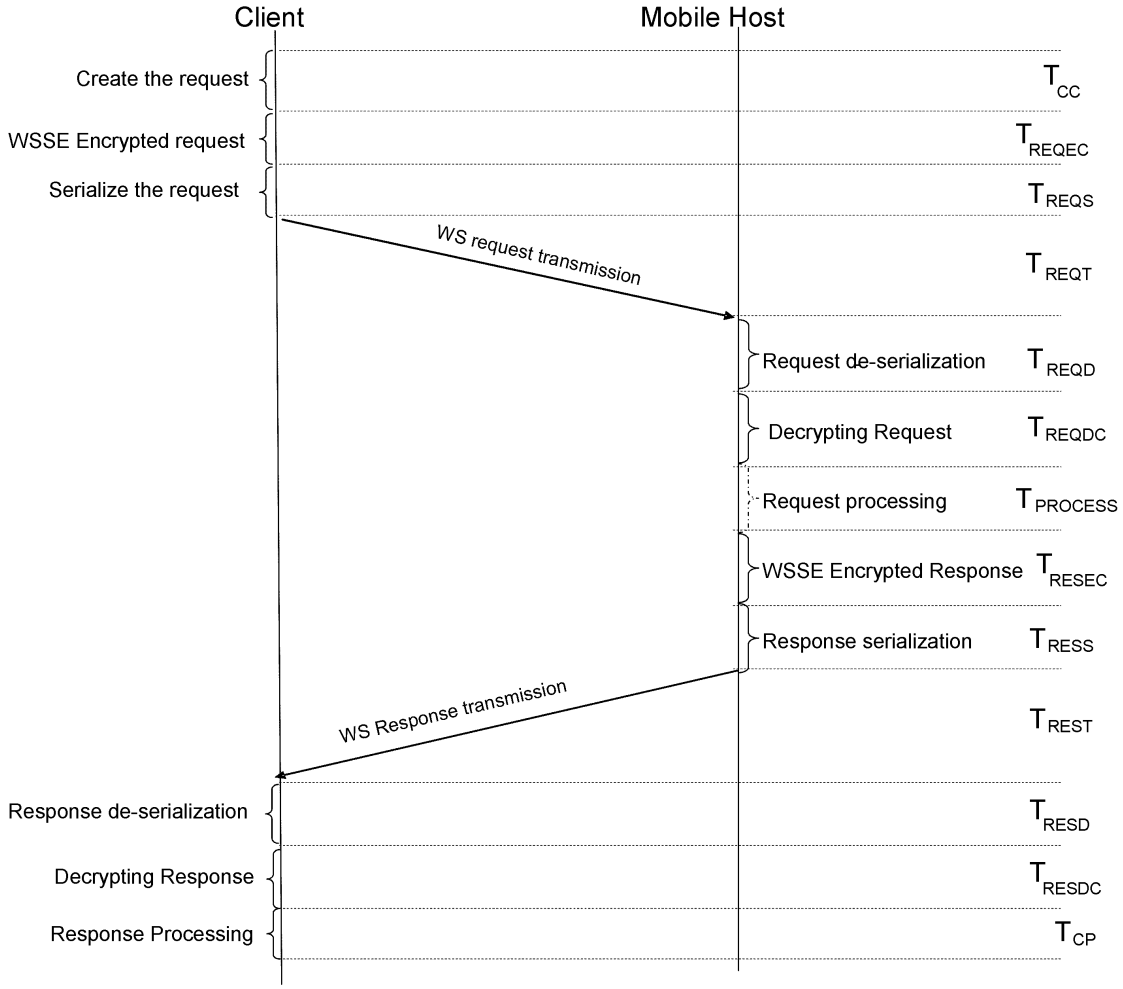


Figure 4.5: Secured mobile web service invocation: Operations and time stamps (Adapted from [Srirama et al., 2007a])

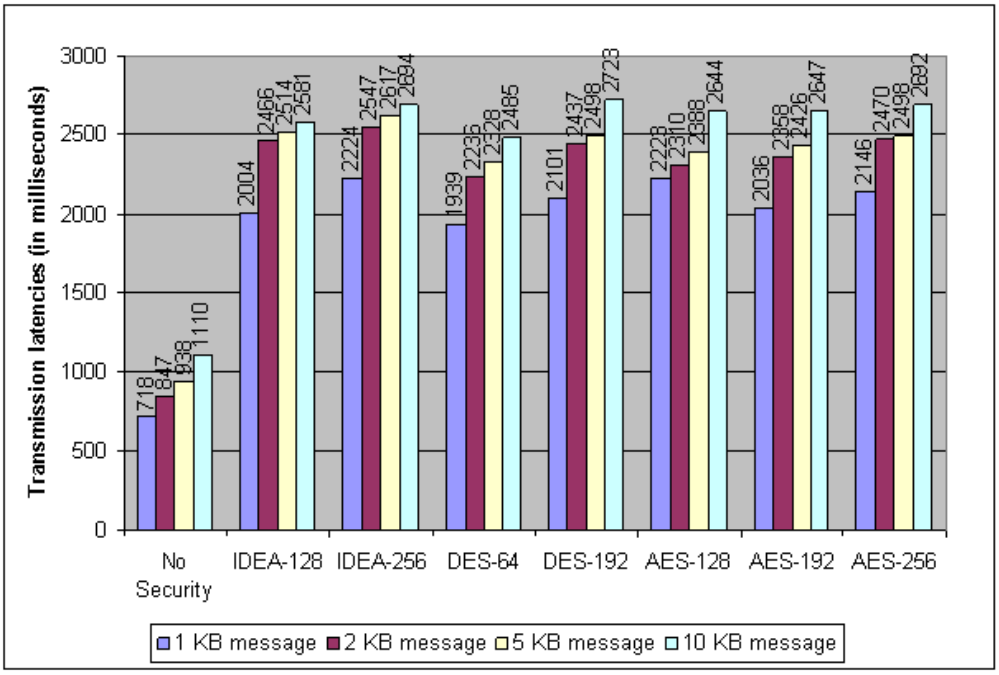


Figure 4.6: Performance latencies with various symmetric key encryption algorithms and exchanging keys with RSA 1024 (Adapted from (Adapted from [Srirama et al., 2007a])

approximately 3 times the latency without any security. The extra delays mainly constitute the times taken for encryption of the request at the client (T_{reqec}), the decryption of the request at the Mobile Host (T_{reqdc}), the encryption of the response at the Mobile Host (T_{resec}) and the decryption of the response at the client (T_{resdc}). From the performance model, we can derive this mobile web service message security effort (T_{mwsse}) as shown in equation 4.2. The approximate equality ($\approx$) is considered in the equation as there is a drastic increase in the size of the secured mobile web service message. This increase in size might have abnormal effects on the performance parameters of the Mobile Host. Unfortunately, the study could not measure these effects, formally. The effects of security incorporation on the size of the message are discussed in section 4.1.6 on page 89.

$$T_{mwsse} \approx T_{reqec} + T_{reqdc} + T_{resec} + T_{resdc} \tag{4.2}$$

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 suggest that the best way to sign the mobile web service message is to use RSA V1.5 with 1024 bit key. RSA algorithm is preferred ahead of the DSA (Digital Signature Algorithm), considering the performance latencies with signing the web service messages, as shown in figure 4.7s.

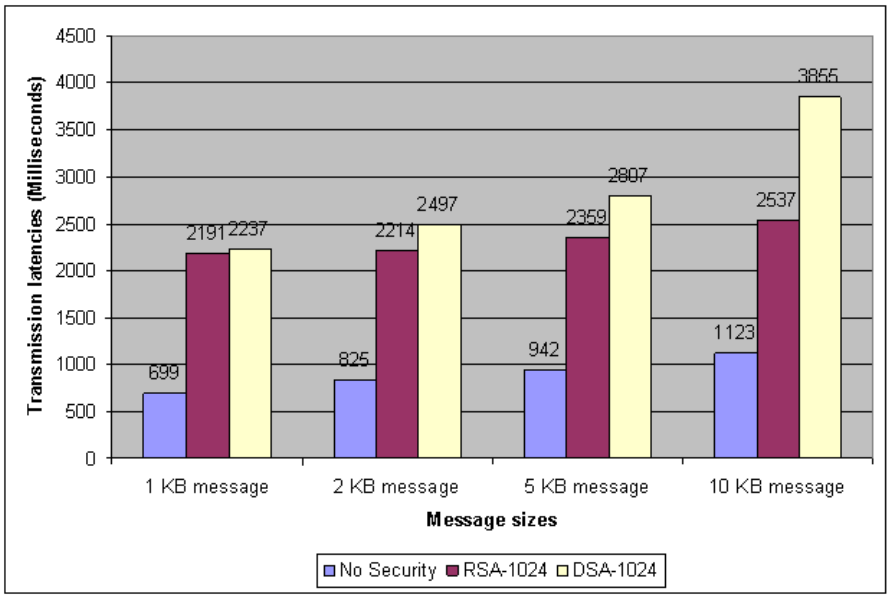


Figure 4.7: Performance latencies with signing the web service messages (Adapted from [Pendyala, 2006])

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, the performance of signing on top of message-level encryption was analyzed in detail. Figure 4.8 depicts times taken for various phases of a message-level secured web service request/response cycle. The timestamps do 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 4.9 on page 90.

From the analysis shown in figure 4.8 and figure 4.9, it can be concluded 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. RSA's consideration for signing is not just based on the increased latencies of DSS, but also depended on the

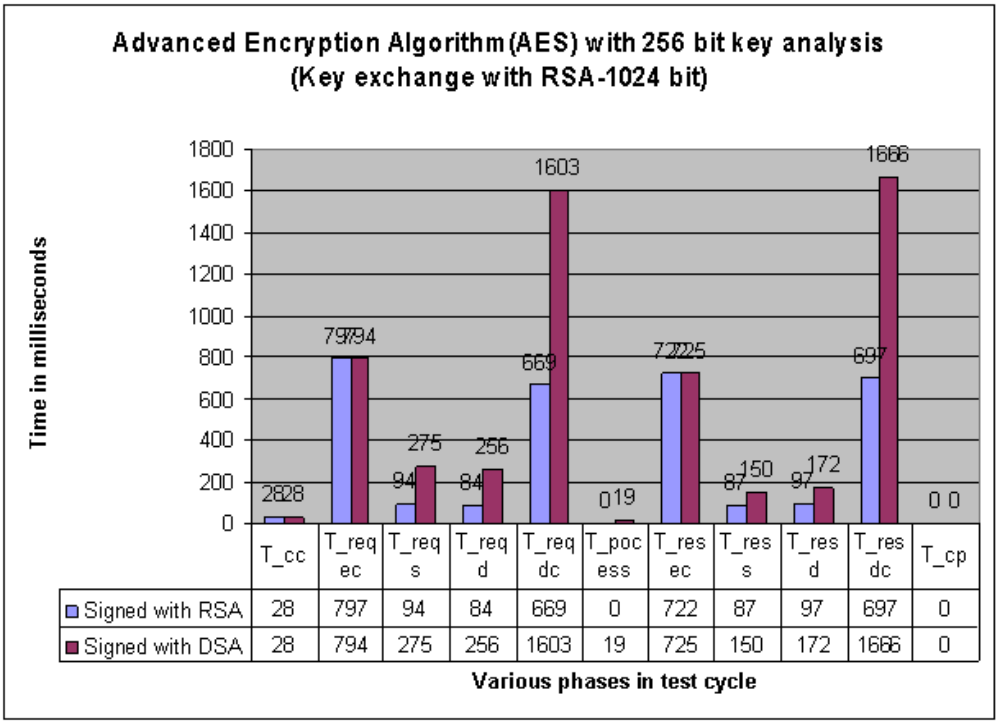


Figure 4.8: Latencies of various phases of a message-level secured mobile web service invocation cycle (Adapted from [Srirama et al., 2007a])

considered asymmetric key exchange mechanism. As the study was using the RSA for key exchange mechanism, the keys can be reused for signing in the complete secured scenario. This saved some initializations and thus reduced the latency in the RSA signature test cases. But there are still high performance penalties when the messages are both encrypted and signed. So the study suggests encrypting only the parts of the message, which are critical in terms of security and then signing the message. The signing on top of the encryption can completely be avoided in specific applications with lower security requirements.

4.1.6 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 listing 4.2. 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 and signing information, is maintained in the SOAP header. The message shown below is the snapshot of a message encrypted with AES, and the key being exchanged with RSA V 1.5. The message was

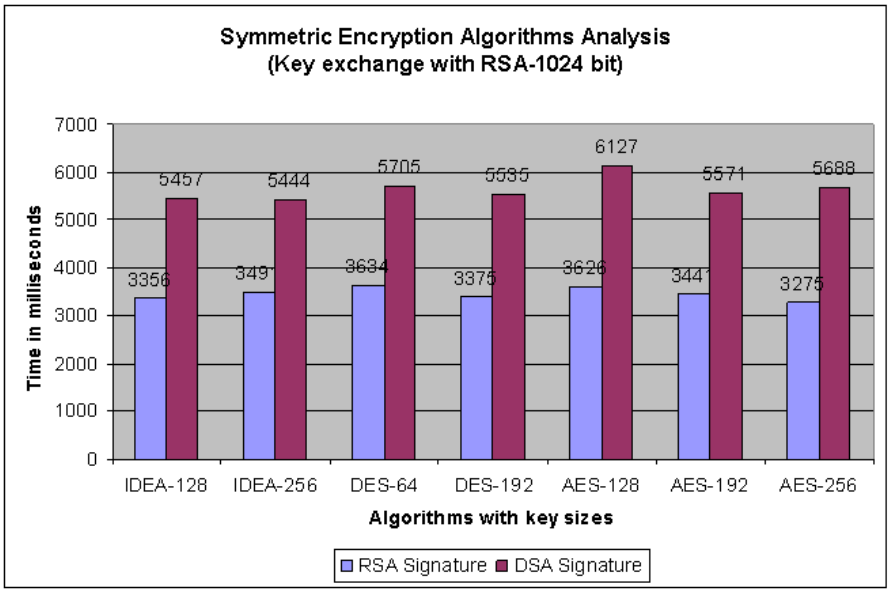


Figure 4.9: Comparison of timestamps for secured mobile web service invocation cycle, across various symmetric key encryption algorithms (Adapted from [Srirama et al., 2007a])

later signed with RSAwithSHA1.

The example showed in listing 4.2 on the facing page also hints the increase in size of the message with the added security header information. Within the security analysis, it was observed that there is a linear increase in the size of the message with the security incorporation. Table 1 shows the variations in mobile web service message size with the applied security. The variation in the encrypted message size for a typical 5 Kb message is approximately 50% [Srirama et al., 2007d]. This increase in size have greatly increased the transmission delays and thus reduced the performance of the Mobile Host. So the thesis studied different means of reducing this size, in the scalability analysis of the Mobile Host, as addressed in section 4.2 on page 106. The study improves the performance of the Mobile Host and thus increasing the host's ability to successfully process more clients (*scalability*).

Original message size	1024	2048	5120	10240
Message size with Signature	1726	2750	5822	10942
Encrypted message size	1804	3168	7264	14092
Secured message size	2611	3975	8071	14899

Table 4.1: Message size variations (in bytes) with security

Listing 4.2: A typical SOAP message incorporated with WS-Security

```
<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>
```

In summary, from this message level security analysis of the Mobile Host, it can be concluded that the security mechanisms developed for traditional networks are not always appropriate for the mobile environment and this area still holds ample room for further research. The results of the security analysis suggest that the mobile web service messages of reasonable size, approximately 2-5kb, can be secured with web service security standard specifications. The security delays caused are approximately 3-5 seconds. It is also concluded from the analysis 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 the study suggests encrypting only the parts of the message, which are critical in terms of security and then signing the message. The signing on top of the encryption can completely be avoided in specific applications with lower security requirements. Thus it was observed that not all of the WS-Security specification can be applied to the Mobile Host. The specification was beyond the resource capabilities of today's smart phones.

4.1.7 Hardware Level Support for Mobile Web Services Security

The results of the message level security analysis suggest that the mobile web service messages of reasonable size, approximately 2-5kb, can be secured with web service security standard specifications. The security delays caused are approximately 3-5 seconds. The delays can significantly be reduced, if there exists a hardware level support for the mobile web service security.

Hardware acceleration [QuickLogic, 2007] is a technique that is commonly employed to get better software performance in terms of speed and hardware accelerators are designed for computationally intensive software code. This approach has been successful with critical processing in domains like graphics, sound processing etc. In these cases, the hardware acceleration offloads the main processing work onto co-processors, also known as *application processors*, and thus frees up the main CPU to do other stuff. This parallelism is one reason we get improved performance out of hardware accelerators. Apart from this hardware accelerators can also provide special instructions that can do the work that is traditionally done by software routines. Thus hardware level approaches for security with proper encryption/signing mechanisms will be quite fast when compared to software approaches.

Apart from the speed of processing, software solutions of any sort cannot facilitate their own physical memory. So they make use of externally available memory, usually through services of the underlying operating system. So there is possibility that other processes can access the same memory space. Although most operating systems give some sort of random access memory space protection, this protection is guaranteed only to the extent of the robustness of the operating system and its being free of flaws.

Cryptographic modules are very sensitive to having their random access memory space

well protected. Most cryptographic algorithms and protocols require intermediate results to be stored during the execution of the secured modules. The temporary stored results can also be the keys. Thus if the contents of this temporary storage are ever leaked the entire system can be easily compromised. Hardware-based solutions can contain their own internally managed memory space, which solves the problem of memory-space protection. Furthermore, hardware solutions can be applied, for memory illegal access prevention. Software solutions are also prone to reverse engineering and power analysis attacks [Bar-El, 2002]. Hardware solutions also consume less power, thus improving the battery life of the smart phones, which is very critical factor in providing proper QoS. But hardware solutions have their own limitations with their high costs and the extra costs incurred with their complete replacement when modifications are suggested at the algorithm level.

Hardware level security for mobile devices is not a completely new concept. Approaches like ARM TrustZone technology [ARM, 2007] and Discretix CryptoCell technology [Discretix, 2007] are some attempts at realising this hardware level security for smart phones. ARM TrustZone technology is a key enabling technology, targeted specifically at securing products such as mobile phones, PDAs, set top boxes or other systems running open operating systems, such as Symbian OS, Linux and Windows CE. TrustZone technology ensures reliable implementation of security critical applications and services such as network virus protection, m-commerce transactions and the protection of user secrets such as keys. The technology is implemented within the microprocessor core itself, enabling the protection of on and off-chip memory and peripherals from software attack. Since the security elements of the system are designed into the core hardware, security issues surrounding proprietary, non-portable solutions outside the core are negated. In this way, security is maintained with minimal impact to the core area or performance of the device. The technology also allows to build any desired additional security features, such as cryptography, onto a secure hardware foundation.

Similarly, products incorporating the Discretix solutions are designed with an open architecture that enables an application provider to build any security application rapidly and effectively. The products are supplied with SSL/TLS, WTLS (Wireless Transport Layer Security), and IPSec (IP Security) protocols together with various security algorithms. Custom applications can be built with a library that accesses the firmware directly to use the basic cryptographic functions as building blocks. Discretix's CryptoCell technology offers a scalable, flexible and highly secured solution for mobile devices. CryptoCell is an advanced embedded security engine, which resides within the base-band chip or the application processor of the mobile phone.

CryptoCell design is divided into different layers. *Hardware layer* provides hardware modules that accelerate asymmetric and symmetric encryption algorithms. *Firmware layer* provides a cryptographic firmware library that provides access to the asymmetric (RSA, DSA etc.), symmetric (AES and DES) as well as hash (SHA-1/2 [NIST, 1995], MD-5 [Rivest, 1992]) algorithms. The firmware layer can be used by upper-layer applications by means of standard cryptographic APIs (e.g. PKCS#11) or proprietary APIs [Elbaz,

2002]. Hence the security for mobile web services can be implemented as a library accessing the firmware layer of Discretix CryptoCell technology on smart phones. The approach benefits from the speed and secured key storage of the hardware platform and thus secured mobile web services can be provided from smart phones with reasonable performance latencies. The approach is left for future research directions in this domain.

4.1.8 Ensuring End-point Security for Mobile Web Services

For providing proper end-point security for the mobile web service provisioning, the basic service-level authentication and user-intervened authorization, discussed in section 4.1.2 on page 79, were observed to be not sufficient. The approaches required additional features at Mobile Hosts, in ensuring the end-point security and thus increased load on the already constrained resources of the smart phones. Hence the study tried for alternative mechanisms for authentication and authorization of mobile web services.

The primary design considered in the study was to analyse the *Single Sign-on (SSO)* to provide both authentication and the possible authorization activities for the Mobile Host. The basic participants required to analyze the SSO scenario are the *mobile web service client*, *Mobile Host*, *Third-party Identity validator*, and *Identity provider*. The SourceID Liberty Beta toolkit [SourceID, 2004] is used for the communication between third party identity validation component and the identity provider. The SourceID Liberty 2.0 Beta is a Java application developed on JBoss application server [JBoss, 2007]. The toolkit, also referred to as ID-FF1.2 (Liberty Identity Federation Framework) Java Toolkit, allows developers build Federated Identity Management services into existing web service projects. Using this open source Java toolkit, we can achieve SSO and federated identity exchange by means of Liberty ID-FF v 1.2 protocols [Cantor et al., 2007]. The toolkit provides generic high-level functionality for user-friendly development by shielding the developer from the complexities of the Liberty protocol, SAML and other dependencies.

In this analysis first the *federation* is established between the mobile web service client and the identity provider. In this federation process, first the client authenticates with third party identity validator and requests for federation with the identity provider which it supports. The credentials are then exchanged with the identity provider according to the SAML standard. The identity provider later requests the client to validate its credentials again to federate into the domain. Upon the respective successful handshakes, the identity provider maintains the federated identity of the mobile web service client. After the federation process, for establishing SSO, two kinds of communication are possible. In the first scenario the mobile web service client initiates the SOAP request with the Mobile Host, without the security token and in the second scenario with the security token. Both the communications are shown in figure 4.10 on the next page.

When a mobile web service client, requests a service from the Mobile Host, the request will be parsed to check the security token for authentication. If the security token is not available, the Mobile Host requests third party identity validator, a standalone component which has mutual trust with Mobile Host, to authenticate the client using Single Sign-on.

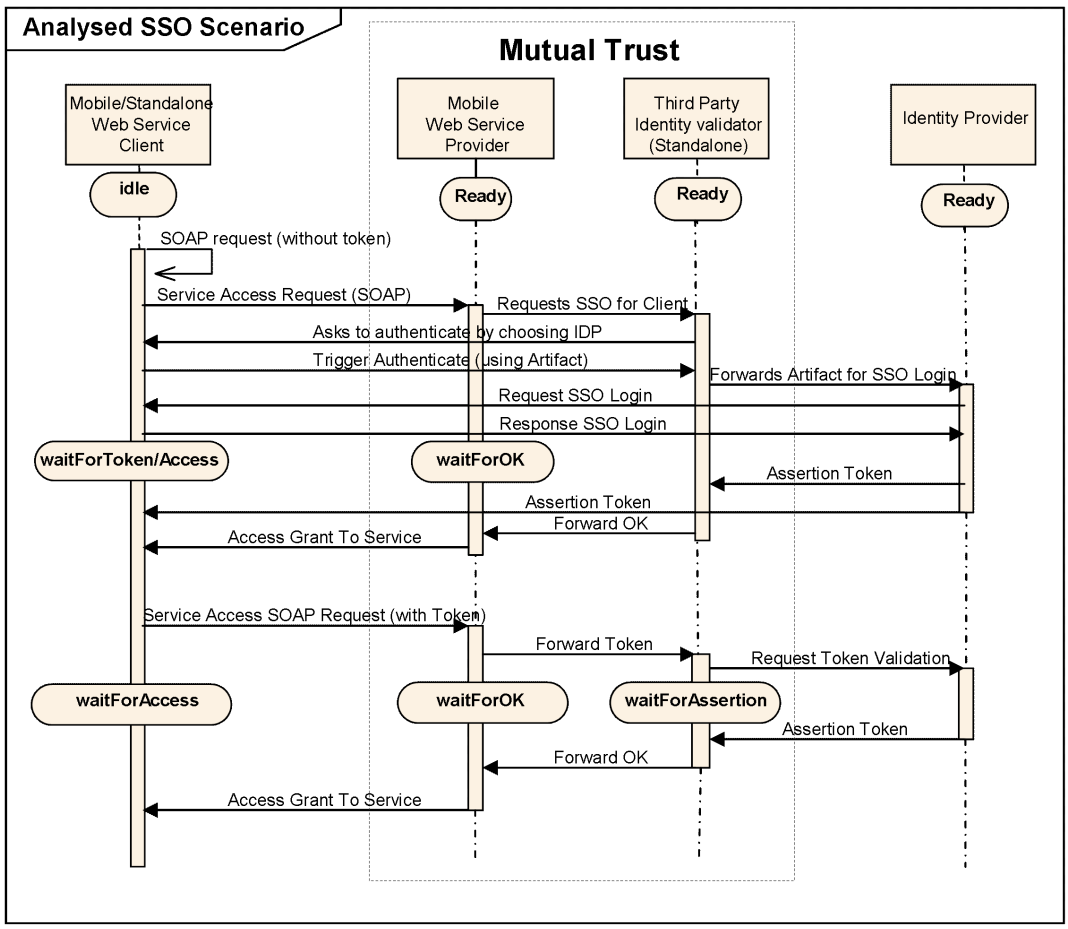


Figure 4.10: Analyzed single sign-on scenario in mobile web service provisioning domain (Adapted from [Pendyala, 2006])

The third party component will then request the client to choose the relevant identity provider with which it is federated. Upon receiving the artifact from the mobile web service client, the third party component forwards the artifact to the identity provider. The identity provider then requests the Single Sign-on credentials by providing its login page to the client. After successful verification of client credentials at identity provider, an assertion token will be sent back both to the client and third party component which forwarded artifact. The client uses the received assertion token along with further service requests which conforms Single Sign-on. Meanwhile, the third party component sends a confirmation to the Mobile Host about the authenticity of the client's SOAP request. The Mobile Host then grants access permission to mobile web service client for its concerned web service.

The second communication scenario for SSO i.e. the request with assertion token is comparatively simple. The mobile web service client requests the service by attaching

the security token to the SOAP request. The Mobile Host then forwards the token to third party identity validator for token validation. At third party component, the token is validated with the identity provider for its authenticity. The component then sends a confirmation to Mobile Host, upon successful validation. The access to the service is then granted for the client.

The scenario was only partially realized, drawing out the architecture and feasibility of its full-fledged usage. The mutual trust between Mobile Host and the third party component are assumed. Of course, it is even possible to exchange the information between these two mutually trusted components in a secured mode using encryption and digital signatures. Apart from this, as session maintenance is not possible at the Mobile Host, each service request with token from mobile web service client will be validated for the clients' authenticity. The scenario is to be further analyzed for scalability and adapting it to the commercial environment.

Alternate scenarios for ensuring this authentication and authorization are also studied and the next subsection explains the analysis with semantic-based access control mechanisms for mobile web services. The analysis is the result of a collaboration of the author with Prof. Dr. Vagan Terziyan, Industrial Ontologies Group, www.cs.jyu.fi/ai/OntoGroup/, Agora Center, University of Jyvaskyla, Finland (responsible person Anton Naumenko annaumen@cc.jyu.fi).

4.1.9 Semantics-Based Access Control (SBAC)

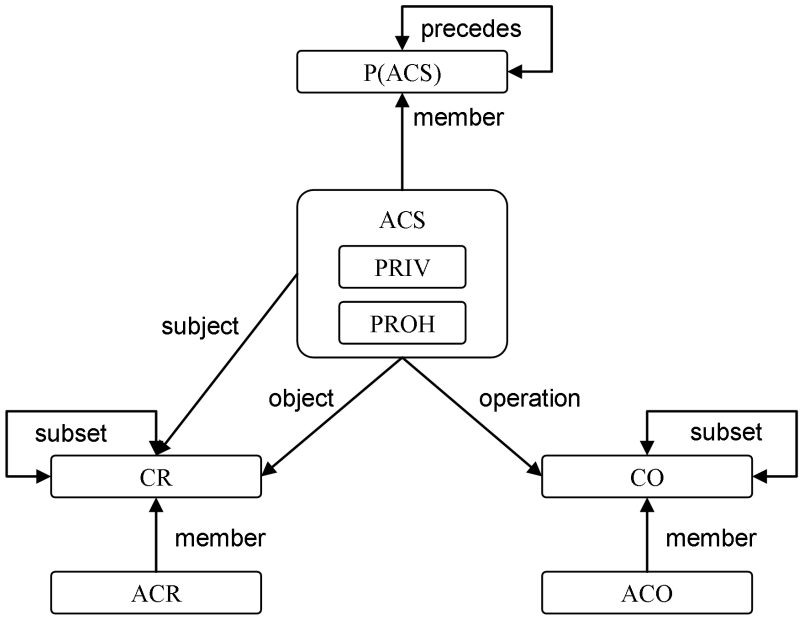
Semantics-Based Access Control (SBAC) is the result of adoption of the Semantic Web vision and standards [Berners-Lee et al., 2001] to the access control research and development field. The *access control* preserves confidentiality and integrity of information in the information systems. The authorization faces the problem to describe (*policy*) rights and enforce (*mechanism*) a decision of access control. It is desirable to use an access control mechanism for a wide range of policies and to enforce one policy in wide variety of environments using native access control mechanisms. An access control model is a mathematically precise statement of a security policy that represents the state of a security system and transitions from one state to another state. Access control mechanisms implement access control models that glue security policies with access control mechanisms. SBAC encompasses administration and enforcement of access control policies based on semantics of web services, clients, mediating actors, contextual data and domain concepts. This is the most suitable approach to handle openness, dynamics, mobility, heterogeneity, and distributed nature of the mobile web service provisioning [Srirama and Naumenko, 2007].

The SBAC Conceptual Semantics

The model-theoretic semantics of SBAC [Naumenko, 2007b] is an extension of the direct model-theoretic semantics defined in the Web Ontology Language (OWL) standard [McGuinness and van Harmelen, 2004] and Semantic Web Rule Language (SWRL) [Horrocks et al., 2004]. The *SBAC model* is a result of introducing vocabularies and interpretations of specific security-related concepts inheriting all features of OWL and SWRL due to the correspondence to their direct model-theoretic semantics. The SBAC model has been expressed in the form of ontologies. Thus, *SBAC policies* are OWL ontologies. The ontology engineering constitutes the traditional domain modeling. The SBAC ontologies consolidate and formally specify knowledge of the access control research and development domain. Similar to the traditional access control models, the SBAC ontologies aim to support the formal specification of policies with respect to standards, legal regulations, domain practices, agreements, approaches, traditions, etc. Ontologies also define languages. The SBAC ontologies define the SBAC policy language. Figure 4.11 on the following page shows the core part of the SBAC model.

ACR is a set of individual resources. A resource is an entity of physical or digital world that is a subject or an object of access. Definition of the resource as a set for subjects and objects gives more flexibility in access control rights specification. Humans and intelligent applications are subjects of access to mobile web services. Domain and policy ontologies engineers annotate and classify resources that are possible subjects of access. Web services have two distinct types of results. A web service can return data or/and affect some real world objects. Generally, SBAC must protect objects of both the types. [Naumenko and Luostarinen, 2006] have studied this heterogeneity of access objects in the context of semantic web services. The considerations and decisions for semantic web services stay the same for mobile web services. The study decided to define access control statements based on inputs and outputs of mobile web services to represent protected objects. Inputs indirectly determine both information objects to be returned and real-world objects to be affected. Access control statements on outputs cannot always be verified before the invocation of services. On the other hand, access control decisions cannot be made after the actual invocation of services in cases when services have effects on real-world objects bundled with outputs. Thus to support the maximum levels of applicability, usability and flexibility, there is a need to provide *pre-authorisation* and *post-authorisation* mechanisms. ACO is a set of individual operations that could be actions, transactions, access modes, etc. Each web services may have several operations. The WSDL operation is the lowest granularity level modelling concept that denotes operations used by subjects accessing protected objects in SOA. Thus, the WSDL operation is the most appropriate concept to be considered as an access control operation. For semantic web services, the concept of atomic process plays the role of the operation.

CR is a set of subsets of resources. CO is a set of subsets of operations. Resources and operations are classified and collected to named sets. CO and CR sets can be partially ordered by the transitive subset relation. This forms hierarchies of resources



ACR, ACO, ACS;
CR $\subseteq$ P(ACR), CO $\subseteq$ P(ACO);
subject:ACS $\rightarrow$ CR, operation:ACS $\rightarrow$ CO, object:ACS $\rightarrow$ CR;
PRIV $\subseteq$ P(ACS), PROH $\subseteq$ P(ACS);
precedes:P(ACS) $\rightarrow$ P(ACS);
access: ACR $\times$ ACO $\times$ ACR $\rightarrow$ Boolean
access(s, o, ob)=($\exists$ priv $\in$ PRIV, s $\in$ subject(priv), o $\in$ operation(priv), ob $\in$ object(priv),
((precedes(PRIV,PROH) $\vee$ precedes(PROH,PRIV), $\neg$ ($\exists$ proh $\in$ PROH, s $\in$ subject(proh),
o $\in$ operation(proh), ob $\in$ object(proh))))

Figure 4.11: The SBAC model: Graphical and textual representations (Adapted from [Naumenko et al., 2007b])

and operations. ACS is a set of access control statements that denotes a many-to-many abstract relation between subject, operation and object of access. The study used three binary relations from access control statements to subject resources, operations, and object resources. The main feature of the access control statement semantics is that these statements are specified between classes instead of individuals. PRIV is a set of privilege statements. It is a subset of ACS. A *privilege* is an authorisation of resources to access other resources using some operations. A decision of access granting or prohibiting depends on memberships of subject, operation and object elements in sets that are in definitions of privileges. The decision algorithm evaluates the containment relation (member) between individual elements and sets taking into account partial order of sets. PROH is a set of prohibition statements. It is a subset of ACS. Support of only positive authorisations in the form of privileges guaranties a conflicts free specification of access control policies. However, even in this case the model has an implicit prohibition that everything is prohibited unless it is privileged. For opposite example, block lists in mobile phones prohibit accepting calls from given phone numbers while there is a general implicit privilege to accept calls from everybody. Introducing means for the specification of prohibitions in the SBAC model enhances expressivity of the policy language. Policies with privileges and prohibitions are not free from conflicts in an arbitrary case. These policies require mechanisms to resolve conflicts and ambiguity. Following the fundamental principle of security, safer is better, prohibitions always precede privileges. Although in the most cases policies will follow the fundamental principle, there is a need to explicitly specify the precedence between privileges and prohibitions. This precedence between sets of access control statements is modelled with a binary relation "*precedes*".

Interpretation of ontologies is the key issue for evolution, consistency, reasoning and organising features of SBAC, domain knowledge and concrete policies separately in different ontologies. This is needed for the flexible knowledge reuse with the high conceptual granularity. Annotation and ontology properties help to record a history of evolution of SBAC and domain ontologies, policies, trust agreements, etc. The definitions of when and how a collection of ontologies and axioms and facts is consistent and entails an ontology or axiom or fact provide background for reasoning and maintaining integrity of SBAC policies.

The SBAC Enforcement Function

The SBAC functionality is naturally separated to two parts: the run-time authorisation function, also called *enforcement*, and the *administrative function*. The enforcement function controls run-time access of requestors to protected resources, according to ontology-based access control policies, attributes of subjects, objects and operations. The SBAC administration function defines mechanisms of manipulation with SBAC data e.g. semantic annotations of resources and operations, domain ontologies, ontology-based policies, configuration settings for the enforcement function, and other.

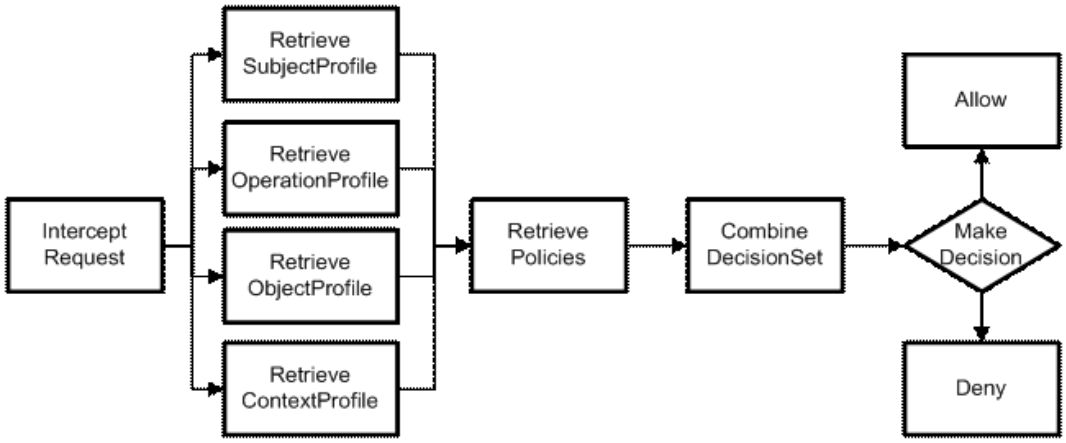


Figure 4.12: The SBAC enforcement function (Adapted from [Naumenko et al., 2007b])

The enforcement function involves several architectural components and nodes. A *subject* of access is a web service client that invokes protected web services to access data or to affect real world. The client accesses the mobile web services on the Mobile Host from a mobile phone or regular computer through Internet and mobile networks. A *guard* mediates access to the protected mobile web service and enforces rules of corresponding access control policies. The guard must evaluate all requests, correctly evaluate semantic profiles and policies, be incorruptible and non-bypassable. A *policy* is ontology. It has access control statements that define which users may access what data using what mobile web services. A subject and object descriptors are well known patterns that provide access to attributes of subject and objects of access. In SBAC, the descriptor pattern is specialised into semantic profiles for users, data, mobile web services, policies and context. Figure 4.12 illustrates a UML (Unified Modeling Language) activity diagram representing the control flow for the enforcement function that is driven by the SBAC guard.

Let us consider a concrete simple example in order to reveal roles of the above described components and roles of activities that comprise the algorithm of the enforcement function. In personal area networks (PAN) users can provide access to information about their location, using location data provisioning service discussed in section 3.1.2. A client sends a SOAP/XML request over Bluetooth to a mobile web service in order to get location data of a provider. A guard intercepts this request. After that, the guard initiates the process of request evaluation. The guard extracts from request's header a URI of web service's operation and supplied credentials of the user. These are input parameters for the activities of retrieving or creating of semantic profiles for the user and operation. The web service's operation does not have any input parameters and always provides the current location only. Thus, there is no need to retrieve the semantic profile of object of access. However, for a more generic case, the guard can create or retrieve the semantic profile for the object based on the input SOAP/XML massage and its WSDL description. After

collecting semantic profiles of the user and operation, the guard retrieves applicable access control statements. In the example, this retrieval uses information about memberships of the user and operation to classes, which were used for specification of privileges and prohibitions. The semantic profiles of the user and operation, retrieved policies, SBAC ontologies are loaded to a decision set for a further decision making. The provider grants privileges to all users that try to access location information using the mobile web service over Bluetooth protocol. There is also a prohibition for a set of users (blacklist) to access the service. This set is specified by enumeration of Bluetooth IDs for some of users. The provider specifies that prohibitions precede privileges. Assuming the user of our example is not in this blacklist, the decision making activity allows this access, otherwise the access is denied.

The architectural components of the SBAC enforcement function are to be deployed to the Mobile Host and middleware nodes depending on characteristics and requirements of usage scenarios. There are several reasonable options of deployment of the SBAC components for protected mobile web service provisioning with unique characteristics and implications on the level of security and quality of mobile web services. Figure 4.13 on the following page shows these deployment options.

The embedded guard (option a) is the most appropriate option for the pervasive mobile web service provisioning within the P2P usage scenarios between mobile clients and Mobile Hosts. The clients directly access services. SOAP/XML messaging between the embedded guard and services can be done using procedure calls without delays of wireless or wired asynchronous communication. One crucial advantage of this option is the opportunity to perform post-authorisations i.e. procedures of access control that must be performed after service enactment e.g. filtering of the content of response. This option supports the principle of end-to-end security in contrast to other options. However computational limitations of mobile phones demand the nomadic functionality of the guard. This undermines the possibility to use complex semantics-based algorithms of reasoning for the embedded authorisation decision making process.

The deployment option b) illustrates the middleware guard that is an intermediate web service proxy. This guard provides the same interface as the original mobile web service, decorates web service invocation with the SBAC policy enforcement mechanism, and delegates authorized requests to the mobile web service. The middleware guard is deployed in the Internet or in the proprietary mobile operator infrastructure. When the guard is in the Internet, clients are able to access it in the traditional way. Moreover Mobile Hosts receive less number of requests or, in other words, only authorized requests. This improves the scalability of the Mobile Host. The post-authorisation is still possible. The middleware guard can represent several Mobile Hosts and web services. Mobile-to-mobile requests experience delays of wireless communication twice when the guard is not embedded but is a middleware component. An additional component of the Mobile Host has to validate security assertions of the guard. The validation of security assertions is necessary in order to check that a security assertion is consistent with a request.

The deployment option c), where the guard is a third-party authorisation authority,

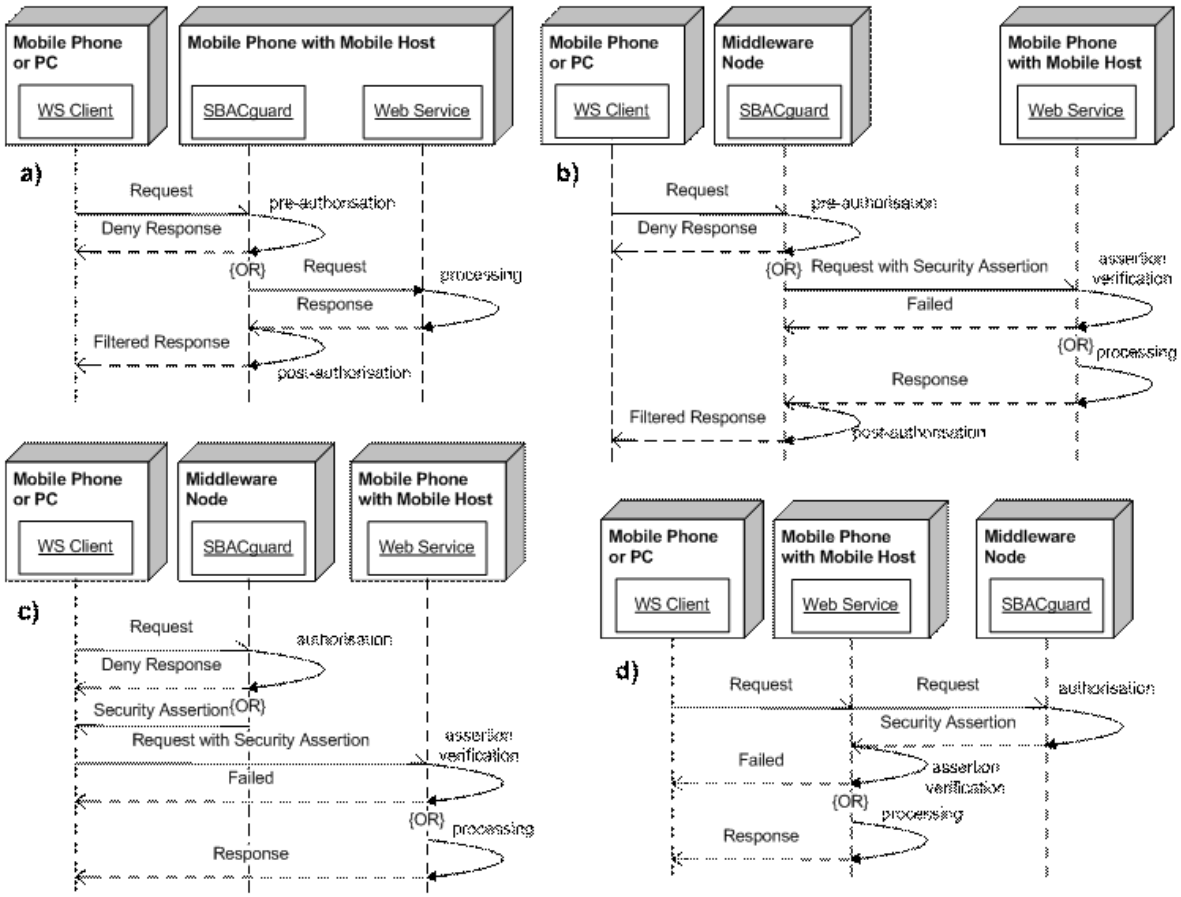


Figure 4.13: SBAC deployment options (Redrawn from [Srirama and Naumenko, 2007])

creates additional inconveniences for clients. They have to get authorisation assertions prior to access protected mobile web services. Then the component deployed on Mobile Hosts validates security assertions provided with requests like in the previous option. Although this case might look too complex, however this is probably the most suitable option for the industrial, commercial or professional use of mobile web services when clients can get security tokens with long period of validity on the basis of their memberships in or subscriptions to different organisations, social networks, commercial services, etc. This option allows direct multiple requests to mobile web services using the same security token over time without overheads of the authorisation decision making process for each request. This deployment scenario is theoretically similar to the SSO mechanism discussed in section 4.1.8 on page 94.

Delegation of authorisation of option d) is the last option considered in the study. Mobile web services initially receive all requests directly from clients and then outsource the decision making procedure to the middleware guard. While such kind of deployment is possible, it has several significant shortcomings without clear advantages comparing to the

above described options. There are following needs: to embed the enforcement component for authorisation messaging with all possible time overheads; to verify signatures of the guard; to process all requests from clients; and other. One advantage is that performance demanding functionality is executed by the middleware guard [Naumenko et al., 2007b; Naumenko, 2007a].

Prototyping and Experimental Evaluation

[Naumenko, 2007a] piloted the decision making procedure to test its performance (makeDecision activity in Figure 4.12 on page 100). This prototype implementation is suitable for the middleware guard as it is, as shown in figure 4.13 on the preceding page. The prototype has to be modified to fit limitations of Java virtual machine of mobile phones, as the framework was initially developed using Java 2 standard edition development kit version 1.5 [Sun Microsystems, 2007b] as a programming language. For the management of policies that are OWL ontologies, the study used a semantic web framework for Java Jena [Jena, 2007]. Jena was developed within the HP Labs Semantic Web Programme [HP Labs, 2007]. Jena has also a SPARQL processor ARQ [ARQ, 2007]. SPARQL is a query language for Semantic Web data [Prud'hommeaux and Seaborne, 2007]. Protégé [Protégé, 2007] was used for piloting SBAC ontologies in the RDF/XML exchange syntax of OWL. Protégé is an open source and free editor of Semantic Web data with the number of plug-ins for editing and visualizing OWL ontologies. For the programming and testing performance the study used tools from Eclipse. Eclipse is an open source community [Eclipse, 2007a] that produces extensible integrated development environment (IDE). The Eclipse Test & Performance Tools Platform (TPTP) project consists of four subproject, one of which provides tools for tracing and profiling Java applications for further analysis of their performance [Eclipse, 2007b].

The internal structure of the decision maker has in-memory knowledge base (decision set) in the form of ontology model provided by the Jena framework and the query engine (query processor) provided by the ARQ processor of SPARQL queries. All ontologies were placed into the web server and were accessible via HTTP protocol. The fastest response time of the decision making process corresponded to the simplest policy and domain ontologies. The policy ontology consisted of one class for mobile web service clients with one user, one class for protected objects with one individual and one class for mobile web service operations with one operation. The policy had the only one privilege statement defined using the above described classes. All RDF statements of SBAC and policy ontologies were loaded into the decision set during the start-up process. Thus, there was no need to retrieve and to load additional information when requests arrived. The SPARQL query corresponded to the authorisation rule for policies defined using only privilege statements.

The average cumulative CPU time of the decision making process was 0.827 seconds. The query execution over the decision set took the major fraction of this time. The personal computer used in the analysis was IBM PC with the CPU AMD Athlon XP

3000+, 1 GB of RAM, and OS Microsoft Windows XP Professional version 2002 with Service Pack 2. Further analysis of the SBAC mechanism and adapting it to the mobile web service provisioning domain is left for the future research.

Use Case and Scenarios Planned with SBAC and Mobile Host

Figure 4.14 on the facing page illustrates a use case of SBAC and mobile web services. The scenario starts when the operators 1 and 2, which belong to different companies A and B, notice abnormal behaviour in their subnetworks. The company A owns the core distribution subnetwork that includes the substations 1 and 2. The company B manages the local power distribution starting from the substation 2 and including the substation 3. Both companies send maintenance workers in order to collect information on-site because initial remote fault detection and localization have not produced precise results. The workers 1, 2 and 3 belong to the maintenance crew of the company A. The worker 4 is send by the operator 2 from the company B. The maintenance workers have installed an application "*Mobile Maintenance Multimedia Blog*" that exposes collected material using mobile web services, e.g. Mobile photo album service and Location Data (GPS) Provisioning Service. These services are protected by the middleware SBAC guards and security assertions verification mechanism of Mobile Host.

The workers 1 and 2 observe substation 1 and adjacent network equipment. The worker 3 checks the substation 2. The worker 4 checks the substation 3 and a physical condition of the feeder between the substations 2 and 3. The workers write textual notes, take photographs, go through checklists, check measurements and other. Before hand, operators and workers got the security assertions from their organizational SBAC guards with permissions to access mobile web services of co-employees. Now, the operators monitor on-site collected material using Internet and GPRS connection to the mobile terminals of corresponding workers (technical usage scenario 1). The workers 1, 2, and 3 jointly browse collected material using a virtual P2P JXTA network extended to the mobile network (technical usage scenario 3, as shown in figure 5.2 on page 123) and a pure P2P (technical usage scenario 4, as shown in figure 5.2 on page 123) using WLAN (Wireless Local Area Network) and Bluetooth connections when possible. In this situation, the middleware guards act as authorization authorities according to the deployment option c).

While checking the substations 2 and 3 and the feeder in between, the workers 3 and 4 try to access the collected material of each other. Their Mobile Hosts delegate the authorization of incoming requests to the middleware guards according to the deployment option d). The access is granted because both companies have an agreement to share maintenance information about all network equipment on the border of their subnetworks. Finally, the worker 4 discovers the tree that felt to the wires and is the most obvious cause of the abnormal behaviour of the network. The photo with the GPS coordinates goes into the maintenance histories of both companies together with all the measurements related to the abnormal behaviour for further reuse during detection and localization of similar

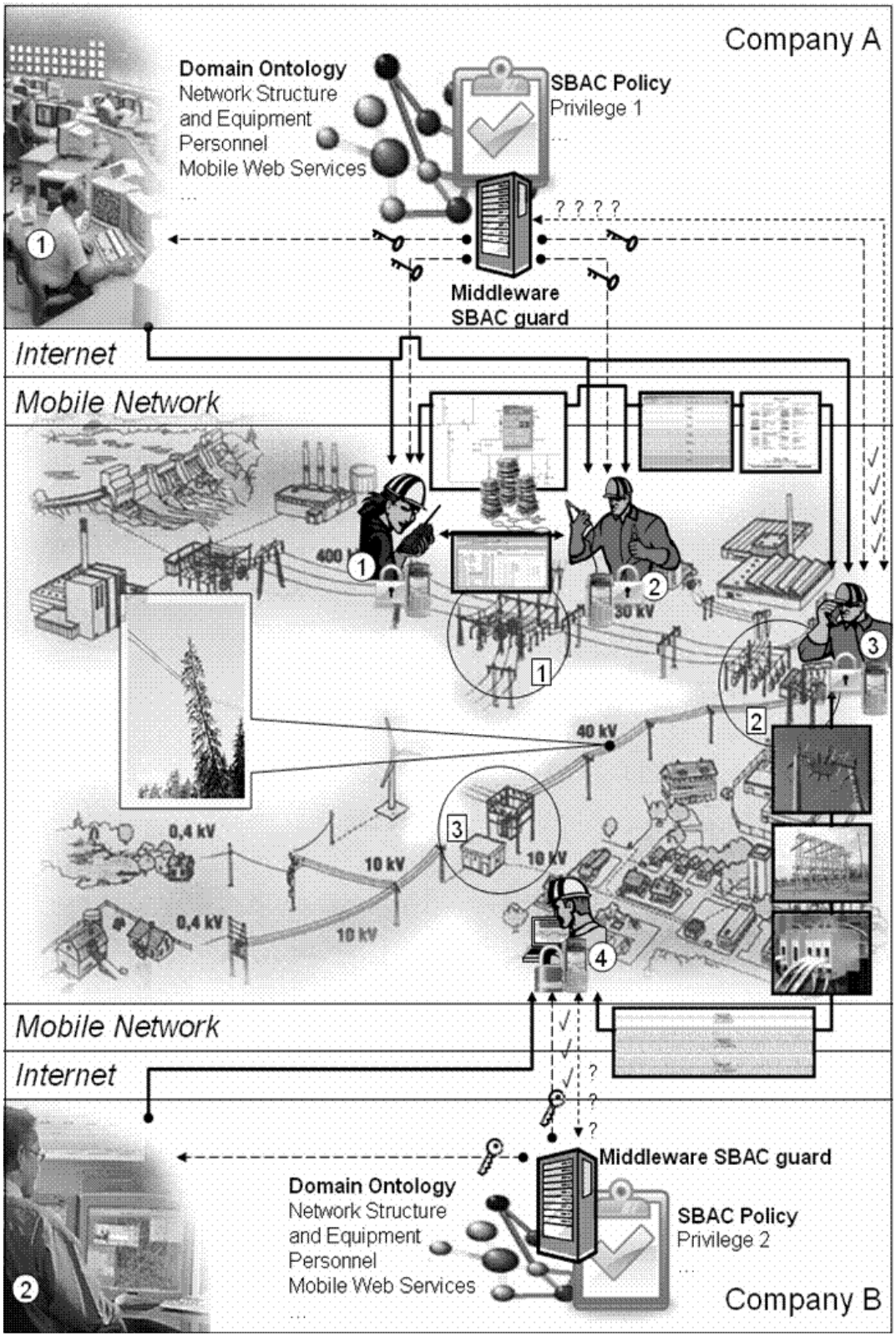


Figure 4.14: Use case of SBAC and mobile web services (Adapted from [Naumenko et al., 2007b])

faults.

In order to exemplify the ontology-based specification of access control policies according to the SBAC ontologies, here the study provides simplified specification of two privileges and related classes in listing 4.3 on the next page. The first privilege authorizes access to the power network equipment of the core distribution subnetwork by the maintenance personnel of the company A using mobile web services, which implement maintenance atomic processes. The second privilege authorizes access to the power network equipment of the substation 3 by the maintenance personnel of the company A using mobile web services that implement maintenance atomic processes. The first privilege belongs to the access control policy of the company A. This privilege is needed by the workers 1, 2, and 3, and operator 1 in order to exchange on-site information. The second privilege allows sharing maintenance information about all network equipment on the border of their subnetworks. In the use case, this second privilege is needed to retrieve the photo with the tree by the worker 3 from the mobile phone of the worker 4. The specifications for these privileges and related classes refer to the SBAC ontologies and to the shared domain ontology "*pwont*". This domain ontology contains semantic annotations about the employees, companies, equipment and structure of power network, mobile web services and other.

4.1.10 Conclusions of the Security Analysis

In the security analysis of the Mobile Host, for ensuring confidentiality and Integrity of the exchanged mobile web service messages a detailed message level security study was performed. The study tried to adapt some of the best principles of standalone web service like WS-Security for the mobile web services domain. The results of this analysis are welcoming and they suggest that the mobile web service messages of reasonable size, approximately 2-5kb, can be secured with web service security standard specifications. The study also proposed hardware level solutions for improving these performance latencies. In terms of end-point security ensuring proper authentication and authorization, the study first realized the basic service-level authentication and user-intervened authorization. Later it tried to realize the single sign-on scenario for mobile web services. Further study of this domain lead to the Semantics-Based Access Control (SBAC) mechanism and its adaptation for mobile web service provisioning domain. In this study different deployment scenarios for the SBAC based middleware are proposed and the prototype developed, which is fully based on open source and freeware components, showed the applicability of SBAC approach for middleware security guards.

4.2 Scalability Aspects of Mobile Host

Similar to providing secured communication for mobile web services, attaining proper scalability is also crucial in achieving appropriate QoS from Mobile Host. *Scalability*

Listing 4.3: Privileges defined in SBAC

```
<sbacpriv:Privilege rdf:ID="Privilege_1">
    <sbs:subject rdf:resource="#MaintenancePersonnelOfCompanyA"/>
    <sbac:operation rdf:resource="&pwont;#MaintenanceAtomicProcess"/>
    <sbs:object rdf:resource="#PartOfCoreDistributionNetwork"/>
</sbacpriv:Privilege>
<sbacpriv:Privilege rdf:ID="Privilege_2">
    <sbs:subject rdf:resource="#MaintenancePersonnelOfCompanyA"/>
    <sbac:operation rdf:resource="&pwont;#MaintenanceAtomicProcess"/>
    <sbs:object rdf:resource="#PartOfSubstation_3"/>
</sbacpriv:Privilege>
<owl:Class rdf:ID="MaintenancePersonnelOfCompanyA">
    <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:about="&pwont;#MaintenancePersonnel"/>
        <owl:Restriction>
            <owl:hasValue rdf:resource="&pwont;#Company_A"/>
            <owl:onProperty rdf:resource="&pwont;#belongTo"/>
        </owl:Restriction>
    </owl:intersectionOf>
</owl:Class>
<owl:Class rdf:ID="PartOfCoreDistributionNetwork">
    <owl:equivalentClass>
        <owl:Restriction>
            <owl:hasValue rdf:resource="&pwont;#CoreDistributionNetwork"/>
            <owl:onProperty rdf:resource="&pwont;#partOf"/>
        </owl:Restriction>
    </owl:equivalentClass>
</owl:Class>
<owl:Class rdf:ID="PartOfSubstation_3">
    <owl:equivalentClass>
        <owl:Restriction>
            <owl:hasValue rdf:resource="&pwont;#Substation_3"/>
            <owl:onProperty rdf:resource="&pwont;#partOf"/>
        </owl:Restriction>
    </owl:equivalentClass>
</owl:Class>
```

is defined as the Mobile Host's ability to process reasonable number of clients, over long durations, without failure and without seriously impeding normal functioning of the smart phone for the user. From the regression analysis of Mobile Host for checking its scalability, discussed in section 3.1.3 on page 40, Mobile Host was successful in handling 8 concurrent accesses for reasonable service like location data provisioning service with response size of approximately 2Kb. The main reason for not being able to process more mobile web service clients was due to the transmission delay which constituted 90% of the mobile web service invocation cycle time. Similar analysis conducted with the mobile picture service, where the response size is approximately 40 kb, further supported this point. So the Mobile Host's scalability is inversely proportional to increased transmission delays. The transmission delays can be reduced in two ways.
1.) By achieving higher data transmission rates with current generation telecommunication technologies.
2.) By reducing the size of the message.
In the scalability analysis of the Mobile Host, the study mainly concentrated at the second issue i.e. reducing the size of the message being transmitted over the radio link.

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 the thesis considered 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 [Laukkanen and Helin, 2003]. The size of the mobile web service message is shown in equation 4.3.

$$B_{msg} = B_{tp} + B_{mtp} + B_{soap} + B_{app} \tag{4.3}$$

Where B_{tp}, B_{mtp}, B_{soap}, B_{app} are the message overheads over transportation, message transportation, SOAP and application protocols respectively. So to exchange the messages effectively over the radio link B_{msg} has to be minimized. For this the messages are to be compressed/encoded in the optimal way. 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 message size reduced and extra performance penalties added to the devices. 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. If an attempt is made to reduce the overload at B_{tp} or B_{mtp}, the interoperability of the web services is seriously impeded. So the best position to target the encoding process is at the B_{soap} and upper levels. So the XML based SOAP messages are to be compressed. The following subsections explain the scalability analysis of the Mobile Host with the achieved performance enhancements due to the compression of exchanged SOAP messages.

4.2.1 Scalability Analysis of the Mobile Host - Design and Implementation Models

To analyse the effects of SOAP message level compression on the performance of the Mobile Host, the exchanged mobile web services messages are compressed with different XML compression standards. As from section 3.4.2 on page 54 and the analysis provided by [Ericsson and Levenshteyn, 2003], BinXML is a good option to compress mobile web service messages considering compression ratio, processing time, resource usage, and ease of implementation. So the study tried to evaluate the effects of BinXML compression on the performance of the Mobile Host, both in terms of improvements to the performance with the reduced message size, as well as the extra performance loads incurred with the BinXML processing on the Mobile Host. For this analysis the Mobile Host is loaded on to a smart phone and the mobile web service clients tried to invoke the services, deployed with the Mobile Host. The clients tried to send the BinXML compressed SOAP messages to the Mobile Host over the HTTP protocol.

The BinXML enabled *Request Handler* component of the Mobile Host is shown in figure 4.15 on the following page. The Request Handler receives the BinXML encoded mobile web service messages over the HTTP Interface and extracts the BinXML encoded SOAP messages. The Request Handler can identify the messages based on the MIME (Multipurpose Internet Mail Extensions) media type, "`application/soap+binxml`". The encoded messages are then transferred to *Binary Encoder/Decoder* component. The component decodes the SOAP messages using the BinXML API. As already explained BinXML API uses the *SAX parser* from the Java ME web services API or JSR-172 (section 3.4.3). The SOAP messages are then transferred back to the Request Handler, which will later be handled by the *WS Handler* component of the Mobile Host, as explained in section 3.1.1. Similar to the requests, the SOAP responses from WS Handler are again transferred to the Binary Encoder/Decoder component, before sending the response back to the mobile web service client. The component BinXML encodes the response and returns to the Request Handler, which transfers the encoded response back to the client.

To analyse the effects of binary compression for mobile web services, two Sony Ericsson P990i mobile phones were used as web service requestor and the Mobile Host. The phones have an internal flash shared memory of 64 Mb [Jerz, 2006]. The devices support MIDP2.0 with CLDC1.1 configuration. The P910i smart phones used in performance analysis and the security analysis of the Mobile Host were not used, as they could not support the JSR-172, which is required in BinXML encoding and decoding of the messages. Hence to compare the performance results of the scalability, the performance analysis was conducted again using the new P990i smart phones. The two mobile phones were connected to the Internet using GPRS connections.

The services of the *Expertise Finder scenario* are deployed on the smart phones and the mobile web service clients tried to invoke these services. The services are explained in detail in section 7.4.5 on page 171, while discussing the applications of the Mobile Host. Listing 4.4 on page 111 and 4.5 on page 112 show the request and response of

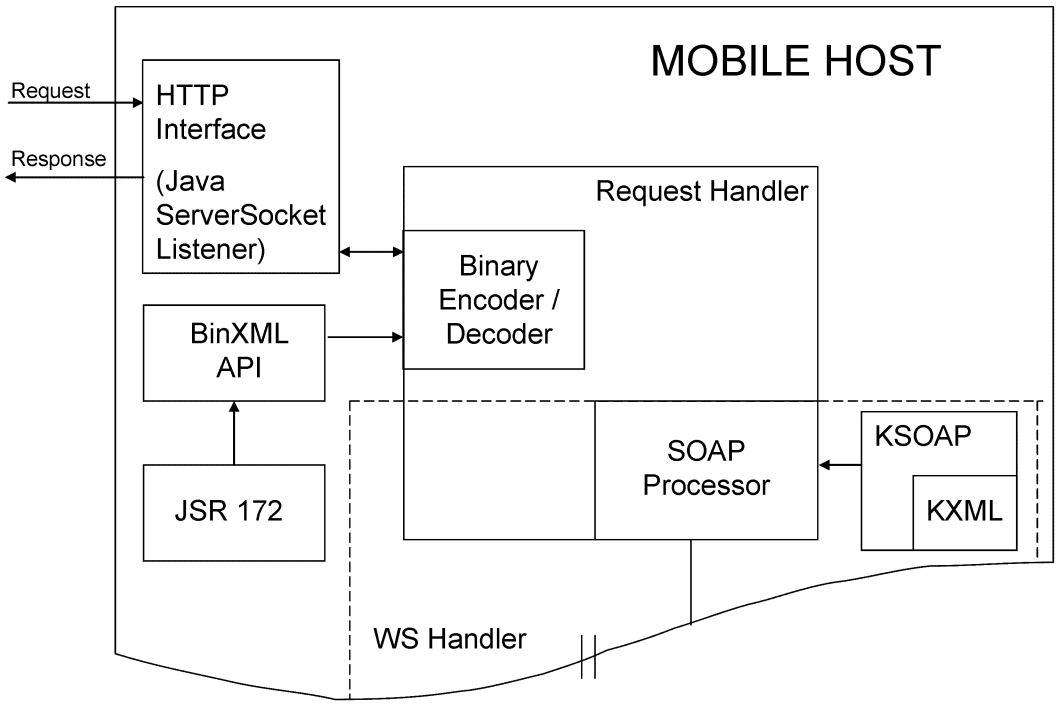


Figure 4.15: BinXML enabled Request Handler of the Mobile Host

Expert Rating service used in the scalability analysis of the Mobile Host. Once the expert is found, in the expert finder scenario, the expert can rate himself to the client, using the expert rating service. The Expert Rating service is considered for the analysis as the messages exchanged with the service have the deeper and repeated XML structures relevant for applying BinXML encoding. The SOAP request message of this service includes the actual expert finder request message (<request>), the intermediaries to whom the message is forwarded before reaching the expert (<forwards>), and the rating and details of the expert (<response>). The size of the request message is observed to be 2544 bytes, with 4 forwards. The response just shows the acknowledgement from the client, and its size is 570 bytes.

4.2.2 Scalability Analysis Performance Evaluation Model

To analyze the performance of the Mobile Host with the compression load, the performance model of Mobile Host, explained in section 3.1.3, was extended to embrace the BinXML encoding intervals. The durations of different activities across the mobile web service invocation cycle are observed. Figure 4.16 on page 113 shows the operations and respective time stamps in the mobile web service invocation cycle. 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. With the added compression parameters,

Listing 4.4: Expert rating service request message

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope ...>
 <SOAP-ENV:Body>
  <mh:expertRating xmlns:mh="http://www.mobilehost.com/" xmlns=...>
   <request xsi:type="mh:request">
    <requestor xsi:type="mh:learner">
     <phone xsi:type="xsd:string">++4917623580216</phone>
     <ip xsi:type="xsd:string">134.130.122.57</ip>
     <name xsi:type="xsd:string">Iliyana Ivanova</name>
     <email xsi:type="xsd:string">iliyanai@yahoo.com</email>
     <other xsi:type="xsd:string">icq:12341234</other>
    </requestor>
    <title xsi:type="xsd:string">J2ME JSR-75 Permissions in JAD</title>
    <description xsi:type="xsd:string">
        Help me to configure JSR-75 permissions in my JAD file </description>
    <startDate xsi:type="xsd:dateTime">
        Thu May 31 10:25:22 UTC 2007</dtartDate>
    <expiryDate xsi:type="xsd:dateTime">
        Fri Jun 15 10:25:22 UTC 2007</expiryDate>

    <forwards xsi:type="SOAP-ENC:Array" SOAP-ENC:arrayType="mh:forward[1]">
     <forward id="1" xsi:type="mh:forward">
      <forwarderPhone xsi:type="xsd:string">++4924102456201</forwarderPhone>
      <forwarderIp xsi:type="xsd:string">137.226.232.12</forwarderIp>
      <date xsi:type="xsd:dateTime">Fri Jun 01 21:36:54 UTC 2007</date>
      <comment xsi:type="xsd:string">Please help my friend.</comment>
     </forward>
     <forward ... > ... </forward>
     . . .
    </forwards>

    <response xsi:type="mh:response">
     <responder xsi:type="mh:learner">
      <phone xsi:type="xsd:string">++4924102456203</phone>
      <ip xsi:type="xsd:string">137.226.232.31</ip>
      <name xsi:type="xsd:string">Satish Srirama</name>
      <email xsi:type="xsd:string">srirama@cs.rwth-aachen.de</email>
      <other xsi:type="xsd:string">none</other>
     </responder>
     <date xsi:type="xsd:dateTime">Sun Jun 03 09:12:44 UTC 2007</date>
     <rating xsi:type="xsd:int">8</rating>
     <comment xsi:type="xsd:string">I think I can do it.</comment>
    </response>

   </request>
  </mh:expertRating>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Listing 4.5: Expert rating service response message

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope ...>
 <SOAP-ENV:Body>
  <mh:expertRatingResponse xmlns:mh="http://www.mobilehost.com/"
    xmlns=...>
   <expertRatingResult>
    <status>The rating was successfully received!</status>
   </expertRatingResult>
  </mh:expertRatingResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

the total time taken for this mobile web service invocation (T_{mwsp}) constitutes, the time taken by client for constructing valid SOAP message (T_{cc}), the time taken to serialize the request message (T_{reqs}), the time taken to BinXML encode the message (T_{reqbec}), the time taken to transmit the SOAP request to Mobile Host (T_{reqt}), the time taken to BinXML decode the request message (T_{reqbdc}), the time taken for de-serializing the XML based SOAP request message at the Mobile Host (T_{reqd}), the time taken by the Mobile Host to execute the respective business logic and to populate the response ($T_{process}$), the time taken for serializing the response message back to XML data streams (T_{ress}), the time taken to BinXML encode the response message (T_{resbec}), the time taken to transmit the SOAP response back to the client (T_{rest}), the time taken by the client to BinXML decode the response message (T_{resbdc}), the time taken to de-serialize the response at the client (T_{resd}), and lastly the time taken by the client to process the response (T_{cp}). The invocation process is shown in figure 4.16 on the next page and the total time taken for the mobile web service invocation is given in equation 4.4.

$$T_{mwsp} = T_{cc}+T_{reqs}+T_{reqbec}+T_{reqt}+T_{reqbdc}+T_{reqd}+T_{process}+T_{ress}+T_{resbec}+T_{rest}+T_{resbdc}+T_{resd}+T_{cp} \tag{4.4}$$

In contrast to the security analysis of the Mobile Host, the transmission times, T_{reqt} and T_{rest} are crucial in the scalability analysis. Unfortunately, the exact estimation of the T_{reqt} and T_{rest} times is not possible as their calculation process needs the synchronization of time stamps of both Mobile Host and mobile web service client. Moreover, the synchronization is to be achieved at the kVM level. The times at the Mobile Host and mobile web service client are synchronized to the extent of seconds and further precisions were assumed to be synchronous. The assumption is also quite logical if only the total transmission time ($T_{reqt} + T_{rest}$) is considered for the analysis. For clarity, if we assume there is a δt milliseconds as the synchronization lag between the two smart phones, then:

$$T_{reqt} = T'_{reqt} + \delta t \tag{4.5}$$

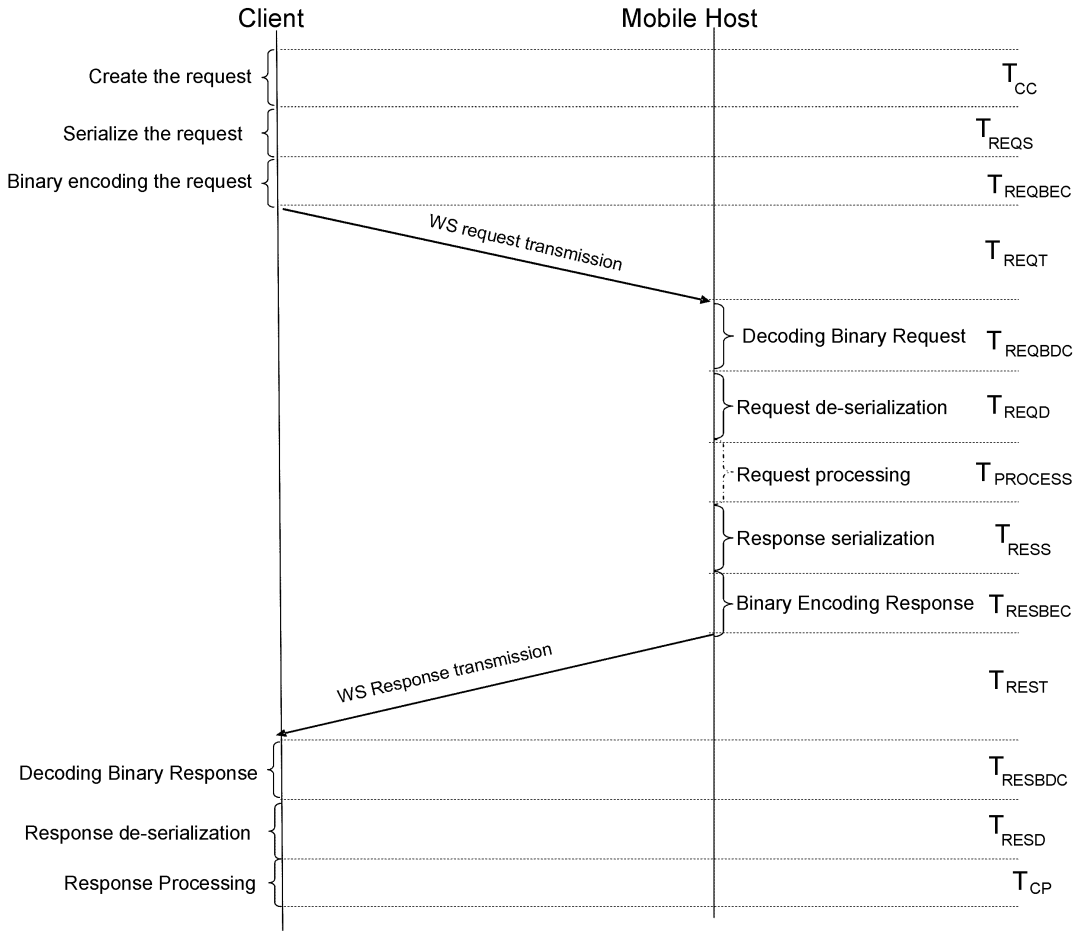


Figure 4.16: Mobile web service invocation with binary encoding: Operations and time stamps

$$T_{rest} = T'_{rest} - \delta t \tag{4.6}$$

Where T_{reqt}, T_{rest} are the transmission delays perceived, and T'_{reqt}, T'_{rest} are the actual transmission delays. Then total transmission time ($T_{reqt} + T_{rest}$) still remains constant, irrespective of the δt value. But the transmission times also included δ_{reqt}, δ_{rest} as the respective propagation delays caused while transmitting the SOAP request and response messages with packet losses, acknowledgements, congestion control etc., as discussed in section 3.1.3. δ_{reqt}, δ_{rest} are network dependent parameters and their exact estimation was not possible. So the tests were conducted multiple times at different instances of the day and the mean values were used to evaluate the effects of binary compression on Mobile Host's performance.

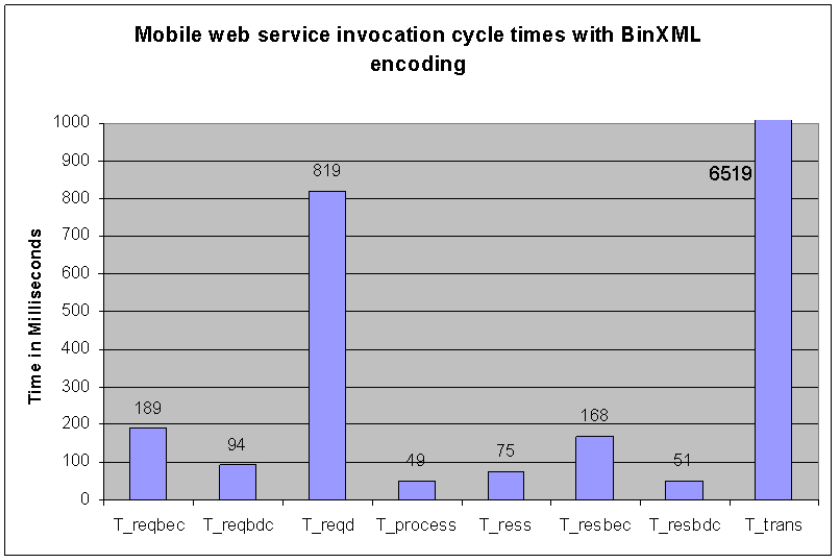


Figure 4.17: Timestamps of different activities for the expert rating service with BinXML encoding

4.2.3 Evaluation of Scalability of the Mobile Host

The main idea of this evaluation is to study the effects of the *message level compression* of mobile web service messages on performance of the Mobile Host. So the study tried to evaluate the effects of BinXML compression on the performance of the Mobile Host, both in terms of improvements to the performance with the reduced message size, as well as the extra performance loads incurred with the BinXML processing on the Mobile Host. The J2ME based Mobile Host was used for this analysis. Since the platform of the Mobile Host changed from PersonalJava, the performance of the Mobile Host was checked again, without QoS effects. These values were later used to draw conclusions from the QoS analysis. The results of this new performance analysis are almost identical to the performance evaluation of the PersonalJava based Mobile Host as explained in section 3.1.3. Still most of the delay is with the transmission.

To analyse the effects of message level compression on the Mobile Host's performance, the expert rating service, discussed in section 4.2.1 is invoked by a smart phone over the GPRS link. Figure 4.17 shows the time delays for different activities with the BinXML encoding added to the mobile web service invocation cycle. The extra delays caused with the BinXML encoding mainly constitute the time taken to BinXML encode the request message at the client (T_{reqenc}), the time taken to BinXML decode the request message at the Mobile Host (T_{reqdec}), the time taken to BinXML encode the response message at the Mobile Host (T_{resenc}) and the time taken to BinXML decode the response message back at the client (T_{resdec}). But with the BinXML encoding, the size of the message to be transmitted over the radio link has reduced significantly. The size of the request

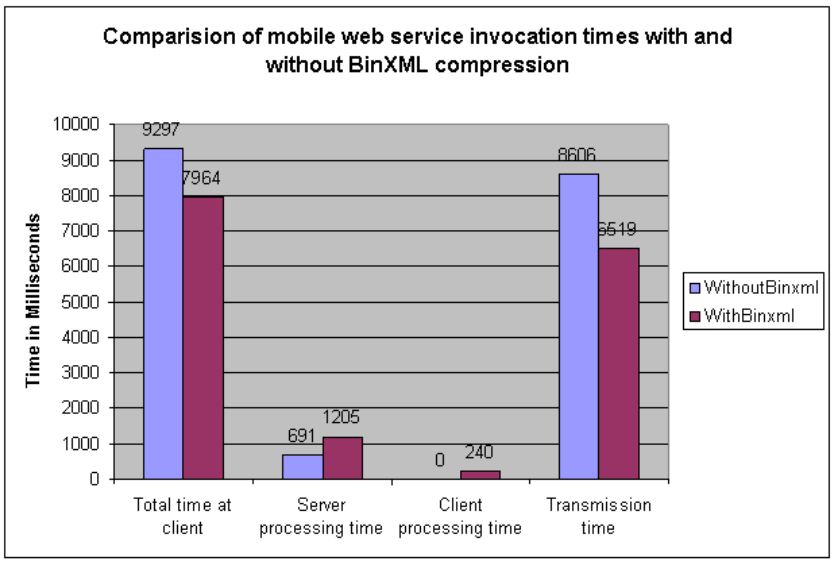


Figure 4.18: Comparison of timestamps for the expert rating service with and without BinXML encoding (Adapted from [Srirama et al., 2008a])

reduced from 2544 bytes to 1591 bytes while the size of the response reduced from 570 bytes to 495 bytes. This reduction in size of the messages to be transmitted has caused significant reduction in transmission delays of the request and response messages. So if we consider δT_{reqt}, δT_{rest} as the respective reductions in request transmission and response transmission delays respectively, then the actual gain in mobile web service invocation cycle time, i.e. mobile web services compression gain with the BinXML encoding is:

$$T_{mwscg} = \delta T_{reqt} + \delta T_{rest} - T_{reqenc} - T_{reqdec} - T_{resenc} - T_{resdec} \tag{4.7}$$

Figure 4.18 shows the comparison of delays in mobile web service invocation cycle, with and without BinXML encoding. From this diagram it can be derived that there is approximately 1333 milliseconds (~15%) gain in performance of the Mobile Host with the BinXML encoding. It can also be concluded that the performance gain is directly proportional to the compression gain achieved with the binary encoding. Apart from this performance gain, since there is significant reduction in transmission delay, the Mobile Host's ability to process concurrent requests has also increased significantly. This has positively affected the scalability of the Mobile Host.

BinXML is very efficient when there is significant repetition of the tags in the SOAP message and the structure of the message is very deep. The result is evident by observing the compression ratios of expert rating request and response messages. The request message has reduced by over 40%, while the response message compressed only by 15%, as it lacked the repetitive tags. Moreover from the results shown in figure 4.17, it can also be observed that encoding times are almost same for both the request and the response

messages. This concludes that even though the BinXML compression scenario is quite efficient for the total invocation cycle; the pure compression times are not significantly different for small messages with few repetitive tags when compared to larger messages. But, as already mentioned, the compression size ratios are significantly different.

Apart from improvements to the scalability, the BinXML encoding also improves the battery life of the Mobile Host. In resource constrained devices like sensors and smart phones, transmitting data in wireless communication is the major energy consumer when compared to CPU cycles [Raghunathan et al., 2002]. So, the lesser the data to transmit, the more power efficient is the application.

Alternative compression mechanisms can also be verified for the Mobile Host's performance gain, with the architecture proposed in this study. As long as T_{mwscg} value is positive, the compression mechanism is efficient. Consequently, with the reduction in size of the message being transmitted, there is a significant gain in the scalability of the Mobile Host.

4.2.4 Conclusions of the Scalability Analysis

Similar to providing secured communication for mobile web services, attaining proper scalability is also crucial in achieving appropriate QoS from Mobile Host. In the scalability analysis of the Mobile Host, it was identified that Mobile Host's scalability is inversely proportional to increased transmission delays. In order to reduce the transmission delays, the study tried to reduce the size of the mobile web service messages being exchanged, using BinXML compression mechanism. From this analysis it can be concluded that BinXML encoding is very efficient in mobile web services domain. The performance gain to the Mobile Host is also quite significant both in terms of improved scalability and battery life.

4.3 Observations for Future Research from the QoS Analysis

The results of the message level security analysis suggest that the mobile web service messages of reasonable size, approximately 2-5kb, can be secured with web service security standard specifications. The security delays caused are approximately 3-5 seconds. The delays are observed to be inversely proportional to the processing capabilities of the today's smart phones. So as the processing capabilities of the next generation mobile phones increase, the delays will be reduced further. The delays can also significantly be reduced with the hardware acceleration support for the mobile web service security, proposed by the study. But even with higher processing capabilities or hardware acceleration support, the analysis produced by the study would still be relevant, and the best scenario observed for securing mobile web service messages will still be apt.

Ensuring the end-point security for the mobile web services is another critical area, which needs to be explored further. The study tried to analyse some solutions like Semantics-Based Access Control (SBAC). The increased processing capabilities might support having functionalities like security guard at the smart phone itself. But only the increase in processing capabilities can not solve the problem in completion. This is an area with lot of scope for future research.

Scalability is another area which will be directly affected by the next generation telecommunication technologies. As the study observed, the Mobile Host's scalability is inversely proportional to increased transmission delays and the transmission delays can be reduced by achieving higher data transmission rates with current and next generation telecommunication technologies. In such high-end networks, the scalability study conducted by the thesis, where it tried to reduce the size of the mobile web service messages being exchanged using XML-aware compression mechanisms, could look obsolete. But the statement would be valid in a scenario where the flat rates in these networks are quite cheap and available for everyone. So as long as the charging in mobile networks is based on volume of data being exchanged, the study is still relevant. The study also helps in avoiding network overloads and congestion troubles.

Moreover, the higher processing capabilities achievable would further reduce the processing delays with compression at smart phones, thus improving the overall gain with compression. This makes the compression solutions and the study of such binary XML solutions further attractive.

4.4 Conclusions

This chapter discussed the quality of service extensions for the Mobile Host. While service delivery and management from Mobile Host are technically feasible, the ability to provide proper QoS, especially in terms of security and reasonable scalability, for the Mobile Host is observed to be very critical. In the security analysis of the Mobile Host the thesis provided a detailed message level security study to adapt some of the best principles of standalone web service like WS-Security for the mobile web services domain. The results of this analysis suggest that the mobile web service messages of reasonable size, approximately 2-5kb, can be secured with web service security standard specifications. The chapter later proposed hardware level solutions for improving these performance latencies.

In terms of ensuring proper end-point security the basic service-level authentication and user-intervened authorization are realized. Further study of this domain lead to the Semantics-Based Access Control (SBAC) mechanism and its adaptation for mobile web service provisioning domain. In this study different deployment scenarios for the SBAC based middleware are proposed and the developed prototype showed applicability of the approach for middleware security guards.

The scalability analysis of the Mobile Host identified that the Mobile Host's scalability

is inversely proportional to increased transmission delays. In order to reduce the transmission delays, the thesis tried to reduce the size of the mobile web service messages being exchanged, using BinXML compression mechanism, observed to be efficient in mobile web services domain. The analysis concludes that the performance gain to the Mobile Host with binary compression is quite significant both in terms of improved scalability and battery life.

5 Mobile Host in P2P Networks

Once the Mobile Host was developed and its feasibility analyzed, as explained in section 3.1 on page 35, the study tried to find specific application domains, where Mobile Host can be adapted. The details of the studied applications of Mobile Host are presented in chapter 7. The study mainly focused at mobile community support and pervasive applications. During this study, it was observed that most of the targeted collaborative applications, somehow converged to *Peer to Peer (P2P)* applications and P2P offered a large scope for several applications with Mobile Host. Not just the enhanced application scope, the P2P technology also offers several technical advantages to the Mobile Host. This chapter explains the analysis with Mobile Host's entry into P2P networks.

5.1 Mobile Web Service Provisioning in P2P Networks

P2P is a set of distributed computing model systems and applications used to perform a critical function in a decentralized manner. P2P networks are typically used for connecting nodes via largely ad-hoc connections. P2P takes advantage of resources of individual peers like storage space, processing power, content, which are all critical for smart phones, and achieves scalability, cost sharing and anonymity, thereby enabling ad-hoc communication and collaboration. In order to reap the benefits of P2P, by achieving increased application scope, and targeting efficient utilization of resources of individual mobile peers, the study tried to adapt Mobile Host into P2P networks.

For this analysis many of the current P2P technologies like Gnutella, Napster and Magi are studied in detail. Most of these technologies are proprietary and are generally targeting specific applications. Only project JXTA [section 3.5.3] offers a language agnostic and platform neutral system for P2P computing. Moreover JXTA community has developed a light version of JXTA for mobile devices, called JXME (JXTA for J2ME). Considering JXME also eliminates many of the low level details of the P2P systems like the transportation details. The mobile peers can communicate with each other using the best of the many network interfaces supported by the devices like Bluetooth, WiFi, GPRS etc. 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 5.1 on the following page shows the architecture of deployment scenario of Mobile Hosts in the JXME network [Srirama, 2006].

As shown in figure 5.1, the *virtual P2P network* also called the *mobile P2P network* is established in the mobile operator network with one of the nodes in operator proprietary

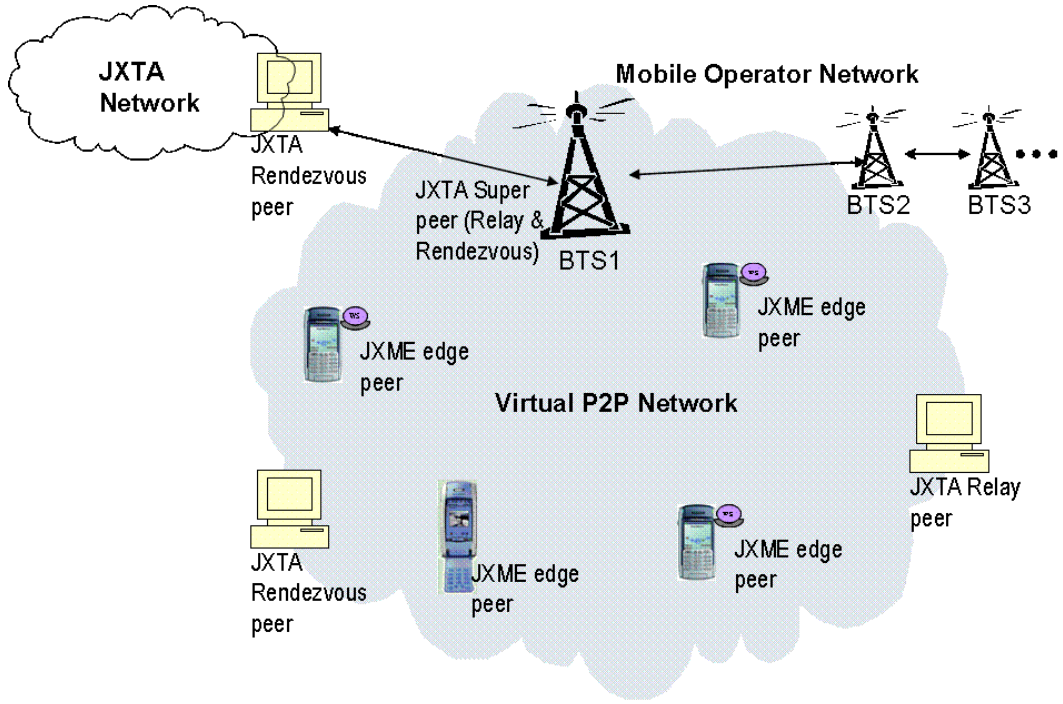


Figure 5.1: Virtual mobile P2P network with Mobile Hosts (Adapted from [Srirama, 2006])

network, acting as a JXTA super peer. As explained already in section 3.5.3, 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 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 several 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 underlying JXTA network, thus eliminating the need for public IP. The necessity of public IP for each of the participating Mobile Hosts was observed to be the major hindrance for commercial success of the Mobile Host. Next subsection explains the analysis with identifying and addressing of Mobile Host [Srirama, 2006].

5.2 Mobile Terminal Access

Once a web service is developed and 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 that 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 connection 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. Very few of the operators provide smart phones with public IP in GPRS network that can be directly used to reach smart phone from Internet. The operational setup for accessing the mobile terminal in a GPRS network is shown in figure 5.2 on page 123 with the interaction numbered 1. 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. The problem of addressing each mobile node with IP is being addressed by Mobile IP version 6 (*Mobile IPv6*) [Johnson et al., 2004]. The key benefit of Mobile IPv6 is that even though the mobile node changes locations and addresses, the existing connections through which the mobile node is communicating are still maintained.

The mobile web service provisioning project also has identified other means of ad-

dressing the Mobile Host in HSCSD (High-Speed Circuit Switched Data) and P2P environments. In the HSCSD addressing scenario, a HSCSD connection is established between the smart phone and the prototyping network, which is connected to the Internet. HSCSD is an enhancement of CSD (Circuit Switched Data) data services of current GSM networks, as explained in section 2.5 on page 31. HSCSD allows the access of non-voice services with a data rate about 3 times higher than that of CSD. Higher rates are achieved by using multiple channels for the data transmission. With this technology subscribers can send and receive data from their portable computers or mobile devices at a speed of up to 28.8 kbps. The HSCSD connection uses a Public Land Mobile Network (PLMN) and the Public Switch Telephone Network (PSTN / ISDN (Integrated Services Digital Network)) 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 [Srirama et al., 2006a]. The interaction is shown in figure 5.2 on the next page with number 2.

As already mentioned, the need for public IP for each of the participating Mobile Hosts was observed to be the major hindrance for commercial success of the Mobile Host. So, alternative architectures were studied for addressing mobile web services. In a 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 the device even though the device supports multiple network interfaces like Ethernet, WiFi or Bluetooth for connecting to the P2P network. Hence, the scope of the Mobile Host in the P2P networks was studied, so as to address the Mobile Host with peer ID. A virtual P2P network can be established by connecting the Mobile Hosts to JXTA superpeers, as explained in section 5.1. Now by using the peer ID, Mobile Host does not have to worry about changing IPs, operator networks, and it is always visible to the web service client. Mapping the peer ID to the IP is taken care directly by the JXTA network, thus eliminating the need for public IP. The JXTA based P2P network also helps in better discovery of huge number of web services possible with the Mobile Hosts, by acting as a dynamic cache of advertisements of the mobile web services. The approach is explained in detail in section 5.3. This Mobile P2P interaction in figure 5.2 on the facing page is numbered 3.

Provisioning of mobile web services in totally decentralized manner is even more challenging. This kind of interaction between peers is also referred as pure P2P. *Pure P2P* is a setup like the classic Gnutella file sharing network. The interactions numbered 4 and 5 in figure 5.2 on the next page represent this pure P2P network idea. In our case of mobile web services, this means that discovery, invocation and integration of web services occur between mobile devices directly without any centralized entities like base stations. The thesis did not study how to provide mobile web services according to this kind of

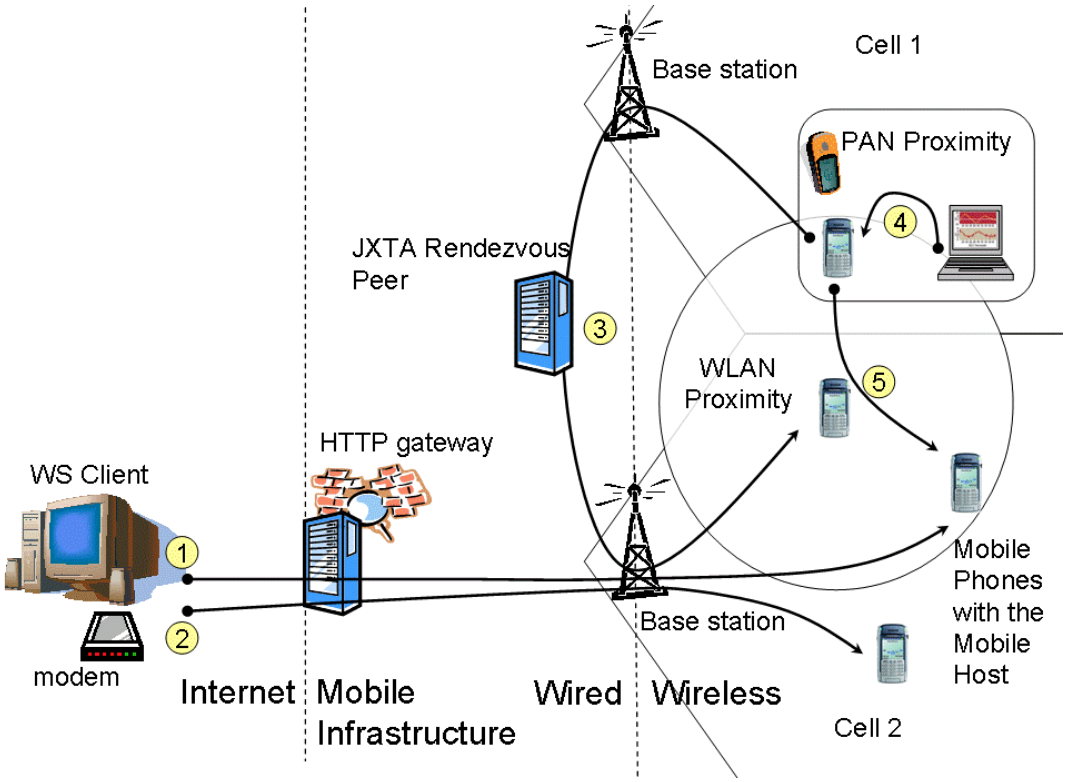


Figure 5.2: Mobile web service provisioning and interactions (Adapted from [Srirama et al., 2008b])

technical usage scenario, but this approach promises to have the best cost-effectiveness as long as interactions between clients and providers of mobile web services do not involve proprietary mobile networks. Bluetooth could be a perfect technical solution for establishing such a pure P2P network. This kind of interactions tends to enable personal computing using various devices in Personal Area Network (PAN) partially or fully based on mobile web services [Naumenko et al., 2007b].

5.3 Mobile Web Service Discovery

In contrast to networked applications, which follow the static install model, mobile applications follow the discover, lease, use, and discard model. So discovery is very crucial and the first step to achieve in deploying applications in mobile environments. Although discovery systems like DNS (Domain Name System) [Mockapetris and Dunlap, 1988; Gulbrandsen et al., 2000], LDAP (Lightweight Directory Access Protocol) [Hodges and Morgan, 2002], and UDDI are available, the systems are designed for statically configured environments, and are not specifically used for mobile computing. The mobile

computing model requires more dynamic, spontaneous discovery features such as mobile, proximity-based service discovery, impromptu community formation, resiliency to faults, minimal administration, etc [Lee et al., 2003]. The following subsections address the discovery aspects of mobile web services and propose the JXTA based mobile web service discovery mechanism.

5.3.1 Discovery Aspects of Mobile Web Services

Typically, web services are built for static networks and are published to a UDDI registry. 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 [Section 2.1.3]. Any potential web service client searching 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, as explained in section 5.2.

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.

5.3.2 Dynamic Service Discovery Mechanisms

Dynamic service discovery is a very extensively being explored research topic. Most of these service discovery protocols are based on the *announce-listen model*. In this model *periodic multicast mechanism* is used for service announcement and discovery. A service advertisement is generally associated with a lifetime during which the service is expected to remain available. Using the advertisements the services are discovered dynamically. This soft-state model is employed for the wireless environments considering the robustness to failure that is likely to happen with the mobile nodes. One of the

frontrunners in this area is the RMI based Jini technology. The Jini discovery mechanism is explained in detail while considering the alternative technologies for Mobile Host development, in section 3.2.

Similarly, [Yang et al., 2003] proposes an infrastructure for organizing and efficiently accessing mobile web services in broadcast environments that defines a *multi-channel model* to carry information about mobile web services. But all these mechanisms assume a service proxy object that acts as the registry and is available always. For dynamic ad hoc networks, assuming the existence of devices that are stable and powerful enough to play the role of the central service registries is inappropriate. Hence services distributed in the ad-hoc networks must be discovered without a centralized registry. Specifically, the discovery protocols should be able to support spontaneous peer to peer connectivity to facilitate ad-hoc collaboration.

[Dustdar and Treiber, 2006] proposes a *distributed peer to peer web service registry* solution based on lightweight web service profiles. They have developed VISR (*View based Integration of Web Service Registries*) as a peer to peer architecture for distributed web service registry. VISR's distributed registry model allows VISR peers to operate on a common data structure and provides a common vocabulary. The approach is context aware and VISR models three types of registry information. 1.) Information about providers (*VISR peer profiles*), that includes data about the web service provider such as memory, CPU, maximum supported web services, name, contact info, etc. 2.) Context information (*VISR view profiles*), that provides context information about web services and serves as means to organize the global web service registry content within web service communities. 3.) Web service descriptions (*VISR service profiles*) that provide a lightweight Web service description. The description is different from WSDL. The idea behind the approach is that the concept of a dedicated Web service registry entity decouples web service providers from storing the actual web service description. So with VISR approach, each node has all the features of a SOA, i.e. service requestor, provider, broker.

Similarly, *Konark service discovery protocol* was designed for discovery and delivery of device independent services in ad-hoc networks. In this approach, a node multicasts the differences between services that a node knows and the ones others seem to know. The messages are called *delta messages*. In other words, each participating node gossips its knowledge about services minus the network's knowledge. So the individual nodes complement one another's limited knowledge to attain the global view of the entire network. For achieving this, Konark is designed based on a P2P model with each participating device having the capabilities to host and deliver services using a resident *micro-HTTP server*, and query the network for available services offered by others [Lee et al., 2003].

In line with these developments Universal Plug and Play (UPnP) [UPnP Forum, 2003] discovery protocol allows devices to advertise their services to *control points* (i.e., clients) on the network. The goals of UPnP are to allow devices to connect seamlessly and to simplify the implementation of networks. Unlike VISR and Konark where each node maintains its own service registry, UPnP does not cache service information at all.

Services multicast their presence announcements (advertisements) periodically. UPnP control points discover services of interest to them either by passively listening to these advertisements or by actively multicasting service discovery request messages.

Considering these developments, and the need for distributed registry and dynamic discovery, the thesis studied alternate means of discovering mobile web services deployed with Mobile Hosts, *mobile web service discovery*. The approach is based on JXTA P2P network and is explained in detail in the following subsections. By following the P2P architecture, mobile web service discovery surely surpasses the problems of announce-listen mechanisms. The approach is conceptually similar to Konark but is truly scalable by following the open standards of web service technologies and P2P protocols of JXTA. JXTA is the most widely adopted P2P technology and it is based on proven technologies and standards such as HTTP and TCP/IP, and is independent of programming language, networking platform, or system platform. Mobile web service discovery still preserves the SOA and the service descriptions are stored only at nodes with higher capabilities (Super Peers). The service descriptions can't be maintained at all participating smart phones because of the resource limitations. At the same time, discarding already discovered advertisements can lead to bottlenecks because of the replica searches. Thus the mobile web service discovery approach significantly differs from VISR where all nodes are registries and UPnP where no node is a registry. The service descriptions also benefit from well known standards of web services technologies.

5.3.3 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 behaviour. Each module class contains one or more module specifications, which contain all the information necessary to access or invoke the module.

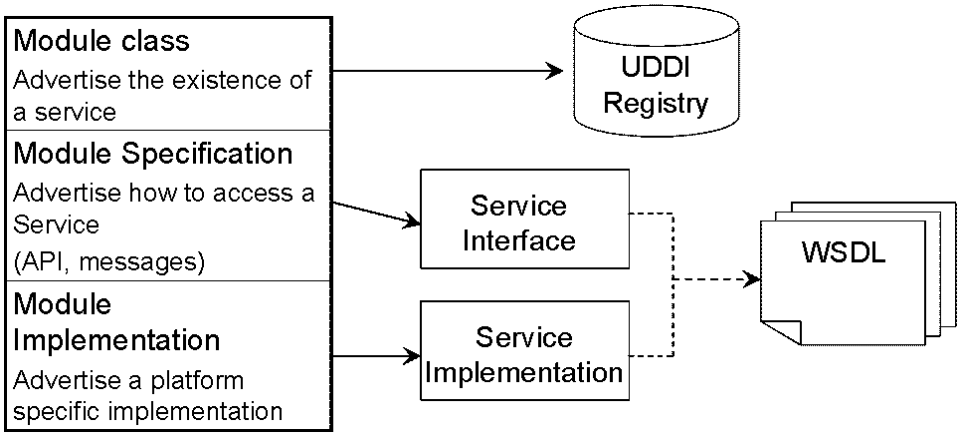


Figure 5.3: Mapping between JXTA modules and web services (Adapted from [Srirama, 2006])

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 5.3 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.

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 particular MSA can be republished. The MSA can be published into the network by a service developer or even by the Mobile Host. Publishing services directly from the Mobile Host is not yet addressed by the thesis. The structure of the MSA is shown in listing 5.1 on the next page.

The MSA contains unique identifier (*MSID*) that also includes the static module class ID, which identifies the web services module class advertisement defining its category. Categories are discussed in section 5.3.5. The other elements of MSID include *name*, *creator*, *specification* and *description* of the advertisement. The optional element `Parm` consists description of the web service (WSDL) 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

Listing 5.1: Structure of an MSA advertising a mobile web service

```
<?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>
```

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 is addressed in section 5.4. 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 web service module.

The module specification advertisements carrying the web service descriptions can be searched by name and description parameters. The JXTA discovery and access model uses a `query routing protocol (QRP)` which is text based, i.e. XML-based [Waterhouse et al., 2002]. Therefore JXTA search implementations are platform and language independent. The JXTA API provides a simple keyword search on the name and description elements of the modules advertised in mobile P2P network. As the study is considering about huge numbers of mobile web services, these basic parameters are not sufficient to find out the exact search results. In fact, some valuable information like context information can not be included in these basic XML tags. Moreover the study tried to extend the search criteria to the WSDL level. This means that search parameters are not 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. A similar approach is taken in UDDI Explorer

tool, developed for searching standalone web services [Dao, 2005]. To handle advanced discovery of mobile web services in P2P, index searching tools are used to match the best suited services.

5.3.4 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 the mobile web service discovery 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 services. Hence only the most relevant of the services should be retuned to the mobile phone i.e. the resulted advertisements should be filtered with weight of the keywords. For achieving this, the study used Apache Lucene tool [Section 3.5.5]. 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, and can index and make searchable any data that can be converted to a textual format. Using the tool and its general-purpose StandardAnalyser, the search results were ordered/filtered according to their relevancy and the *advanced matched services (AMS)* were returned to the discovery client. Lucene makes the deep search using name, description and parm attributes of MSAs.

This detailed search mechanism can not be performed at the JXME edge peer because of the resource limitations of smart phones. The advanced search mechanism can be shifted to a standalone distributed middleware. For this, the study tried to realize an Enterprise Service Bus (ESB) [Keen et al., 2004] based "*Mobile Web Services Mediation Framework*" (MWSMF) [Srirama et al., 2006c], which maintains the individual user profiles, personalization settings and context sensitive information. The architecture and implementation details of the MWSMF are addressed in chapter 6. In this advanced matching of services scenario, the proxy node to which the Mobile Host is connected, participates in the mediation framework, handling the discovery issues. The discovery support of the mediation framework is currently under study and is left for future research directions in this domain.

5.3.5 Categorization of Mobile Web Services

In order to make discovery process for mobile web services more efficient, apart from index filtering the resulted services from JXTA search, the modules advertising the web services in JXTA can also be properly categorized using features like 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 also be maintained in JXTA. The idea here is to achieve some of the best features of UDDI like *categorization* to mobile web service discovery. The ability to attribute metadata to services registered in

UDDI, and then run queries based on that metadata is absolutely central to the purpose of UDDI at both design time and run time. Hence categorization is arguably the most important feature of UDDI. This subsection introduces the UDDI categorization and the conceptual model of achieving this categorization in P2P based mobile web service discovery.

Simple Categories in UDDI

All four main UDDI data structure types, which are *businessEntity*, *businessService*, *bindingTemplate* and the *tModel*, provide a structure to support attaching categories to data. By providing a placeholder for attaching categories to these data structures, the categorization is achieved in UDDI and it can be used for a variety of purposes. The following XML fragment shows an example of how categories are added to a businessEntity using a *categoryBag*. The businessEntity instance contains a categoryBag with two UNSPSC (United Nations Standard Products and Services Code) product category codes and three ISO 3166 (International Organization for Standardization) country codes, and specifies that the company sells medical equipment and pharmaceutical products in Germany, France and the United States. In the example, each <keyedReference> refers to a category in the category system. The tModelKey of each reference uniquely identifies the tModel that represents the category system. Similarly keyName represents human readable name of the category system while keyValue mentions the actual category code within the specified category system. The keyValue can also be simple keywords instead of full-fledged category systems codes [Bellwood, 2002]. Listing 5.2 on the facing page gives an example categoryBag, defining a company's products and locations of operation.

Most of the times, the use of single categories is sufficient for describing the characteristics of a business, a service, a binding, or a tModel so that it can be easily found. But there can be cases where the relationship between single categories becomes important. For example, if the business described in listing 5.2 on the next page, wants to specify that it actually sells medical equipment only in Germany and France and pharmaceutical products only in the United States, then the categories are to be somehow grouped together. For this purposes, categoryBags can contain *keyedReferenceGroups* that in turn contain a list of *keyedReferences*. Since the set of keyedReferences that are grouped within a keyedReferenceGroup do not themselves provide any meaning of the grouping, the keyedReferenceGroup carries its own tModelKey identifying a tModel that in turn provides this meaning. Listing 5.3 on page 132 shows how keyedReferenceGroups are used in order to achieve the desired behavior.

Adapting Categories in Mobile P2P Services

UDDI categorization helps the web services requester to find the needed services more quickly and efficiently because the searching scope of instances could be narrowed immediately with the given standard. The same categorization idea could help us to speed

Listing 5.2: UDDI category bag defining a company's products and locations of operation

```
<businessEntity businessKey="uddi:my_company.example">
    ...
    <categoryBag>
        <keyedReference
            tModelKey="uddi:uddi.org:ubr:categorization:unspsc"
            keyName="UNSPSC:Medical Equipment and Accessories and Supplies"
            keyValue="42.00.00.00.00"/>
        <keyedReference
            tModelKey="uddi:uddi.org:ubr:categorization:unspsc"
            keyName="UNSPSC:Drugs and Pharmaceutical Products"
            keyValue="51.00.00.00.00"/>
        <keyedReference
            tModelKey="uddi:uddi.org:ubr:categorization:iso3166"
            keyName="GEO:Germany"
            keyValue="DE"/>
        <keyedReference
            tModelKey="uddi:uddi.org:ubr:categorization:iso3166"
            keyName="GEO:France"
            keyValue="FR"/>
        <keyedReference
            tModelKey="uddi:uddi.org:ubr:categorization:iso3166"
            keyName="GEO:United States"
            keyValue="US"/>
    </categoryBag>
</businessEntity>
```

up the mobile web services discovery in JXTA environment. Categorization is also critical in terms of resource consumption, in the targeted environment of mobile phones. In order to deploy categorization in to mobile web service discovery, a placeholder is needed to attach information about categories to module advertisements defining the mobile web services. So the category information is added to the Module Class Advertisement (MCA). The category structure is maintained in `<categoryPack>` element. The CategoryPack will be defined and inserted in description of the Module Class Advertisement. The structure of the MCA with the added categorization information is shown in listing 5.4 on page 133. Similar to MSAs, MCAs can be published into the network by a service developer or by any standards agency that defines categories or even by the Mobile Host.

The MCA contains unique identifier (`MCID`) which identifies the module class advertisement i.e. the category of mobile web services. The other elements of MCA include *name*, and *description* of the advertisement. The description of the MCA is incorporated with the `categoryPack`. All the mobile web services that belong to this category, have the MCA as the module class of the MSA holding the advertisement of the mobile web service. As already mentioned, MSID also includes the static module class ID that defines the functionality. The category groups can be maintained with peer groups i.e. all the mobile web services that belong to a particular category group are advertised in the particular

Listing 5.3: A category bag showing the use of `keyedReferenceGroup` in UDDI

```
<businessEntity businessKey="uddi:my_company.example">
    . . .
    <categoryBag>
        <keyedReferenceGroup
            tModelKey=
                "uddi:uddi.org:ubr:categorizationgroup:unspsc_geo3166">
            <keyedReference
                tModelKey="uddi:uddi.org:ubr:categorization:unspsc"
                keyName="UNSPSC:Medical␣Equipment␣and␣Accessories␣and␣Supplies"
                keyValue="42.00.00.00.00"/>
            <keyedReference
                tModelKey="uddi:uddi.org:ubr:categorization:iso3166"
                keyName="GEO:Germany"
                keyValue="DE"/>
        </keyedReferenceGroup>
        <keyedReferenceGroup
            tModelKey=
                "uddi:uddi.org:ubr:categorizationgroup:unspsc_geo3166">
            <keyedReference
                tModelKey="uddi:uddi.org:ubr:categorization:unspsc"
                keyName="UNSPSC:Medical␣Equipment␣and␣Accessories␣and␣Supplies"
                keyValue="42.00.00.00.00"/>
            <keyedReference
                tModelKey="uddi:uddi.org:ubr:categorization:iso3166"
                keyName="GEO:France"
                keyValue="FR"/>
        </keyedReferenceGroup>
        <keyedReferenceGroup
            tModelKey=
                "uddi:uddi.org:ubr:categorization:unspsc_geo3166">
            <keyedReference
                tModelKey="uddi:uddi.org:ubr:categorization:unspsc"
                keyName="UNSPSC:Drugs␣and␣Pharmaceutical␣Products"
                keyValue="51.00.00.00.00"/>
            <keyedReference
                tModelKey="uddi:uddi.org:ubr:categorization:iso3166"
                keyName="GEO:United␣States"
                keyValue="US"/>
        </keyedReferenceGroup>
        . . .
    </categoryBag>
</businessEntity>
```

Listing 5.4: Structure of an MCA advertising a category of web services

```
<?xml version="1.0" encoding="UTF-8"?>
<jxta:MCA>
<MCID> . . . </MCID>
<Name> . . . </Name>
<Desc>
  <categoryPack>
     <keyedReferenceGroup . . . >
        <keyedReference . . . />
        <keyedReference . . . />
     </keyedReferenceGroup>
    </categoryPack>
</Desc>
</jxta:MCA>
```

peer group. The hierarchy of categories can also be maintained with this mechanism. The root of this hierarchy is the default web services group (WSGroup). The study maintained the categorization of mobile web services by observing some of the famous categorization systems like the North American Industry Classification System's (NAICS) [U.S. Census Bureau, 2007] service industries classification. Figure 5.4 shows the default category group hierarchy considered by the thesis. It consists of four levels and fifty-seven peer groups altogether with mobile Web Service Group (*mWSGroup*) on the top level. The group structure is a first draft to realize the idea of categorization. To construct a complete categorization hierarchy for all possible mobile web services is neither realizable for the present stand of the research nor meaningful for the need. The application of the hierarchical categorization structure is implemented by a shell command in JXTA platform. With this self-defined shell command the whole categorization structure is published into JXTA network. The hierarchy is advertised along with the establishment of mobile P2P network and can be provided as a reference manual for the mobile web service discovery mechanism.

5.3.6 Evaluation of Mobile Web Service Discovery

To evaluate the mobile P2P discovery approach, initially, 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 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 5.5 on page 135. The mobile web services developed for 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 5.5, searched

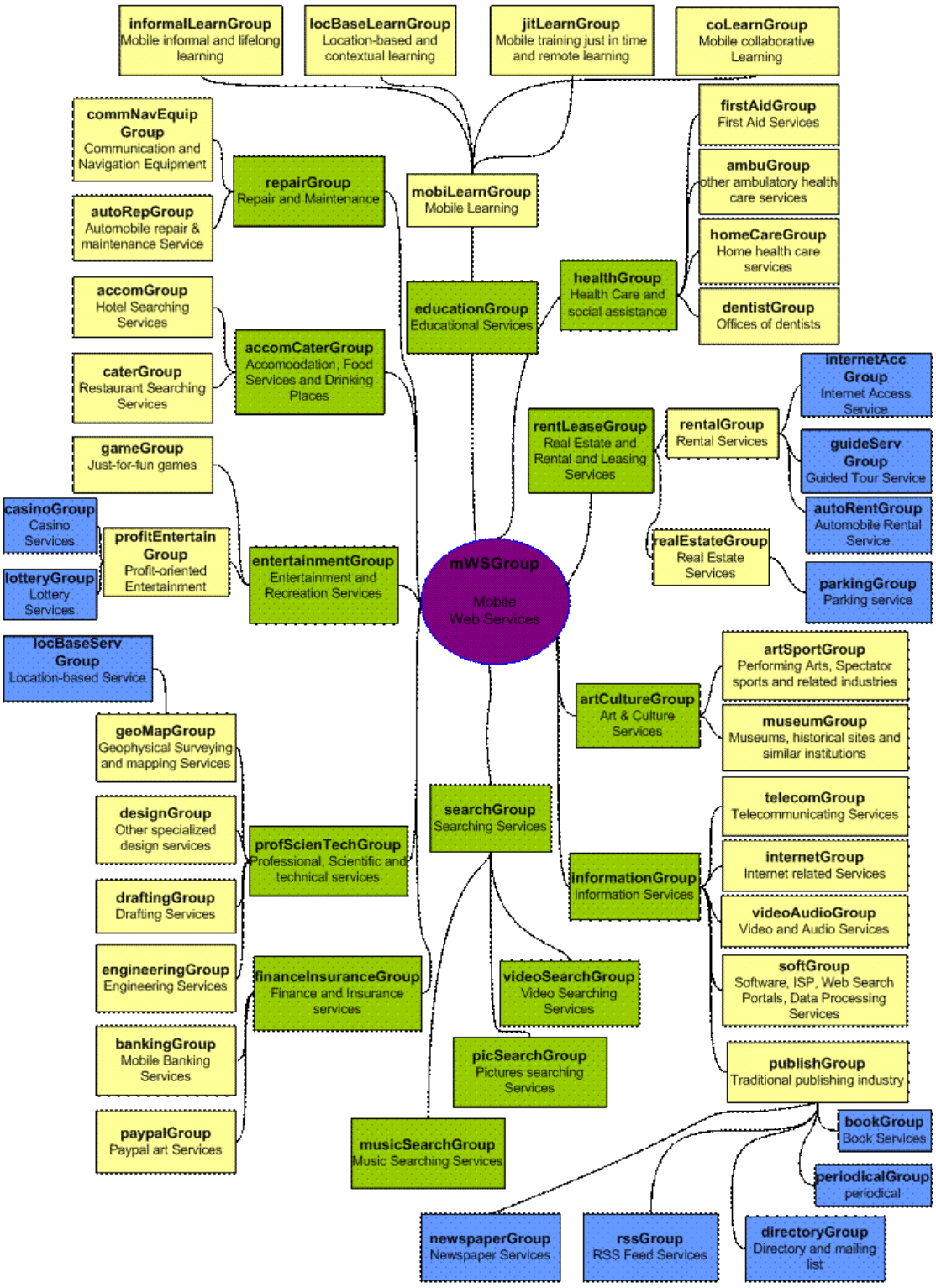


Figure 5.4: Mobile web services categorization hierarchy (Redrawn from [Srirama et al., 2008b])

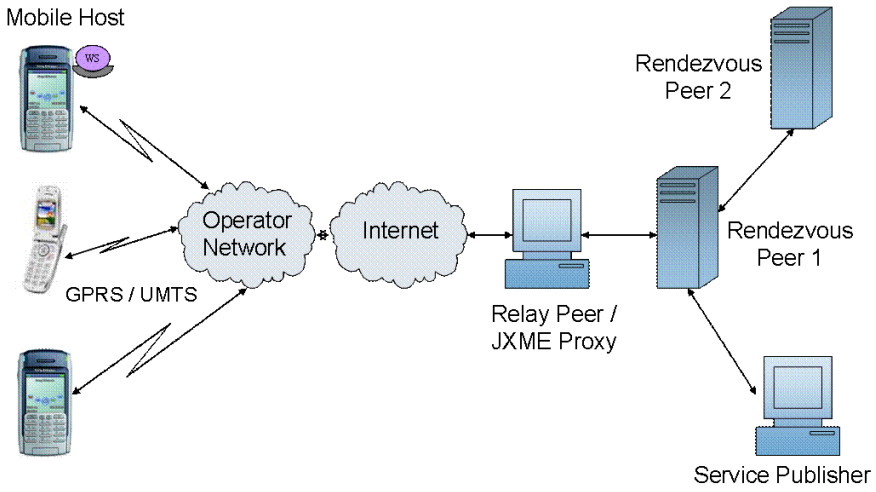


Figure 5.5: The P2P based mobile web service discovery evaluation scenario (Adapted from [Srirama et al., 2007b])

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 [Srirama et al., 2007b].

In order to statistically determine the scalability of mobile web service discovery approach, the study reviewed the related projects. JXTA Benchmarking project is used to measure the scalability of JXTA networks. The goals of JXTA Benchmarking project is to construct a test harness which can be set up by people relatively unfamiliar with JXTA, that operates more or less automatically (scripted), which reports on network performance measures of interest to the JXTA community. Two other projects further implemented JXTA Benchmarking project to some extent, regarding performance evaluation of typical peer operations and sequences [Halepovic and Deters, 2003] and the performance of peerview and discovery protocols by large-scale, multi-site experiments [Antoniu et al., 2007] respectively.

With the above mentioned three projects as reference, the study conducted the scalability analysis of the mobile web service discovery, on the basis of peer operations in the process of discovery under different topologies. In the test, startup benchmark is set as main metric for pre-discovery stage and Round Trip Time (RTT) benchmark is set as main metric for discovery stage. The benchmark suit included four network topologies where the numbers of participating rendezvous peers (RDV) are 0, 1, 2 and 3 respectively. The topologies are shown in figure 5.6. The discovery mechanisms both with and without categorization are considered for the analysis.

The hardware setup for the evaluation included a Sony Ericsson P990i smart phone and a pool of eight computers. The phone has a memory of 60 MB and 3G technology capability with data transfer speeds up to 384 kbps for Internet. But, as mentioned earlier,

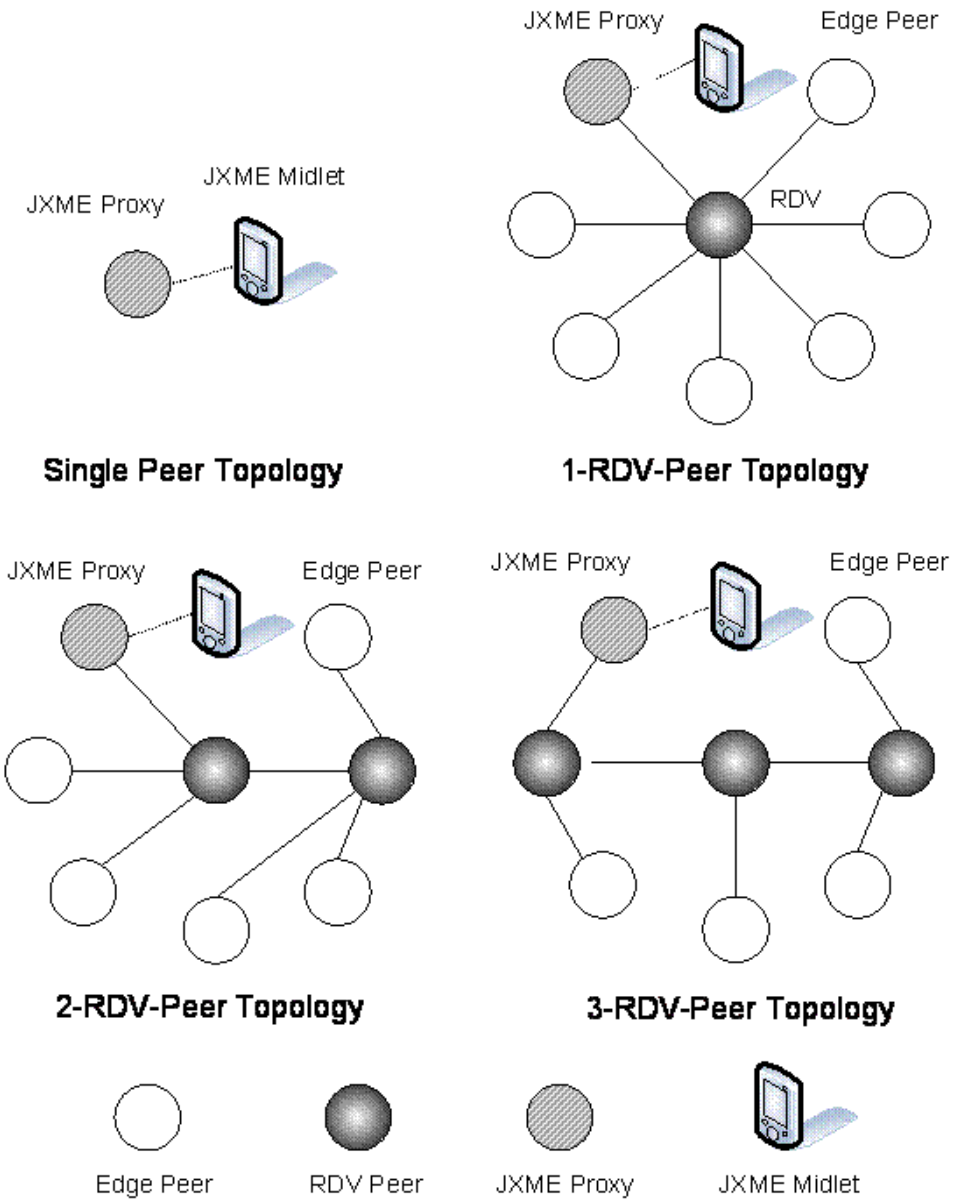


Figure 5.6: Scalability evaluation of the mobile web service discovery - test topologies (Adapted from [Srirama et al., 2008b])

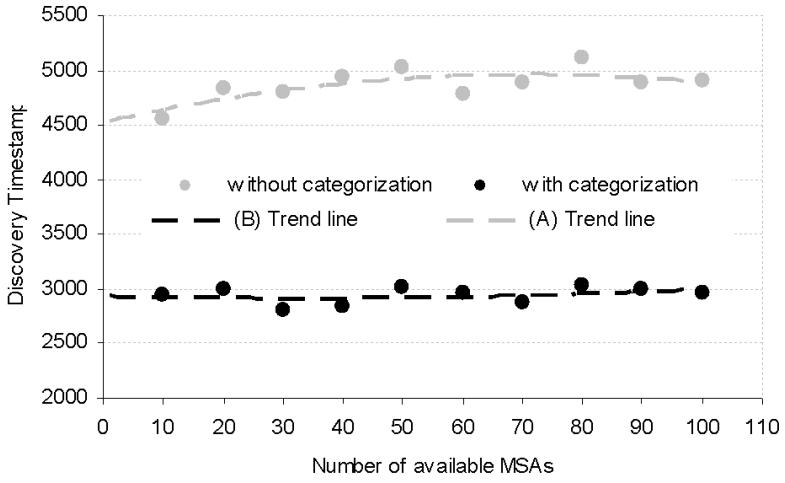


Figure 5.7: Comparison of timestamps for topology with 2 RDV (Adapted from [Srirama et al., 2008b])

the thesis used GPRS connection for establishing access to the Internet from smart phones. The computers are equipped with Pentium IV, 3.2GHz (Gigahertz) and RAM (Random Access Memory) of 1GB or 2GB. The network environment is campus 100 Mbps LAN at RWTH (Rheinisch-Westfälische Technische Hochschule) Aachen, Germany. JXTA-JXSE version 2.4.1 was used to execute the test.

For obtaining the performance results, large no of MSAs (in the range 10-100) are published into the group categorization hierarchy. All the experiments are conducted in all the 4 different topologies, under multiple number of matching services for the keywords. In the case of single peer topology with no RDV, non-categorization discovery is comparatively easier to conduct. Since the only variable is the number of MSAs to be published. The discovery latency is measured in each case of different number of MSAs in the local cache of the single peer. In categorization discovery the measurement is more complicated. Since four levels of peer groups are built in the category hierarchy, as shown in figure 5.4, and with the concern that the discovery latency may be rather diversified on different levels of category hierarchy, the study measured the performance from each level of hierarchy. With number of MSAs to be published as the variable, discovery latency is measured from groups of each of the four levels.

Figure 5.7 shows the results for the topology with 2 rendezvous peers. From these results it can be observed that It costs about 41% less latency for a peer to find an MSA with discovery mechanism with categorization than non- categorization discovery mechanism. The trend line of non-categorization discovery mechanism shows a light linear form, while that of discovery mechanism with categorization keeps rather constant trend line. The mean of categorization discovery is 2872 ms, in comparison to 4866 ms.

From the scalability results it was also observed that the percentage difference for categorization and non-categorization discovery mechanisms in case of topology with no RDV is approximately 50%. Moreover, one or two more RDV on the route of discovery bring no dramatic effect to the discovery time and trend. The results from Two-RDV and Three-RDV topology are very much alike to One-RDV topology. The non-categorization discovery approach keeps the tendency of mild linear growth, while categorization discovery approach leads to almost constant discovery time hardly affected by the growing number of available MSAs in peer groups.

These results indicate that the use of categorization in discovery process obviously improved the performance by 40% to 50% in average. With the addition of more RDV peers, the discovery time of non-categorization version grows slightly, which might lead to undesirable scalable property in large-scale network. In contrast the discovery time with categorization mechanism does not grow much, if at all, with the addition of more RDV peers. Although it is just a small-scale test result, it can still be concluded that the design of mobile web service discovery mechanism with categorization could very probably bring scalable performance even under the large-scale networks [Zhu, 2008].

Even though the mobile web service discovery is technically feasible, and only returned relevant results, the relevancy of the resulting services were observed to be a little skeptical, even after applying categorization. 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. The following subsection discusses the context aware mobile web service discovery analysis.

5.3.7 Context Aware Mobile Web Service Discovery

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 Web Ontology Language (OWL) [McGuinness and van Harmelen, 2004] based Web Ontology Language for Services (OWL-S) [Martin et al., 2004] can be used. OWL-S is an ongoing effort to enable automatic discovery, invocation, and composition of web services.

The context is the information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to interaction between a user and an application including the user and application themselves. Context-awareness is a property of a system that uses context to provide relevant information and/or service to the user, where relevancy depends on the user's task. Thus, context-aware service discovery can be defined as the ability to make use of context information to discover

the most relevant services for the user [Dey, 2000]. The OWL-S ontology is composed of *ServiceProfiles* describing the capabilities of the services like inputs and outputs, *ServiceGroundings* describing the invocation details of the services like communication scheme, address, ports, etc., and *Service-Models* describing the tasks and behavior of the services. The ServiceProfile describes the functional and non-functional aspects of a web service, and is therefore used for mobile web service discovery. The ontologies can later be processed during the service matching [El-Sayed and Black, 2006].

But the semantic discovery process is heavy, in terms of both resource consumption and performance latencies like extra delay. So after the analysis of the discovery approach, the thesis suggests using the P2P discovery mechanism first to reduce the search space. The resulted services (AMS) can then be matched semantically for the most relevant results. The context source provides the context information of the services and web service clients. The web service description contains the reference to its context source. Thus the context information is incorporated into the advertised MSAs. The context engine matches the suitable services for the mobile web service client by processing the context information of advanced matched services (AMS) and the context of mobile web service client itself. The context engine exists as a component at the mediation framework. The collected context information of the AMS will be stored in services graph structure. The context engine uses semantic match making algorithms to obtain the most suitable services. The study planned to use Jena [Jena, 2007] for semantic matchmaking. Jena includes a rule-based inference engine, support for ontologies, a querying mechanism and a persistent storage capability. The context aware mobile web service discovery is left for future research directions in this domain and the study suggested the related components and their deployment locations for realizing the discovery process, in section 6.3 on page 148.

5.3.8 Mobile Web Service Discovery Process

The complete mobile web service discovery scenario, envisioned, is shown as an activity diagram in figure 5.8 on the following page. The client initiates the query for services at the mediation framework. The details of the mediation framework are explained in chapter 6 on page 145. The mediation framework, searches for the matching modules advertisements in the JXTA network. The module advertisements are then filtered with weight of the keywords. The advanced matching services are processed at the context engine, considering the context information of the services and the client. The matching results are then forwarded back to client. The client scrolls through the list of the services and selects the relevant mobile web service. The user can then access the web service from the Mobile Host. Just as a hint, in terms of numbers, the advanced matching of services should return a set of approximately 50 services, of which the semantic matching should reduce the services to a scrollable set (5 - 10) for smart phones. The complete publishing and discovery process of mobile web services in P2P networks is summarized in next subsection.

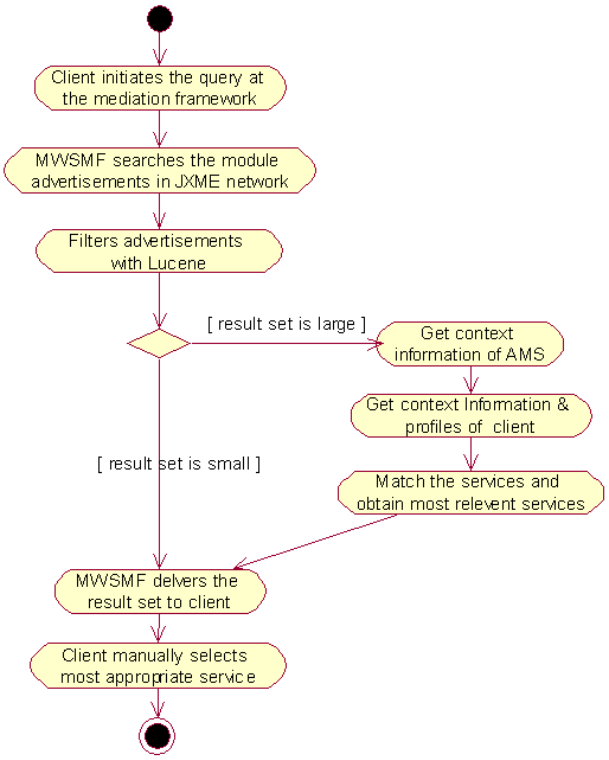


Figure 5.8: Complete mobile web service discovery process

5.3.9 Publishing and Discovery of Mobile Web Services in JXME Networks

Previous subsections have discussed the aspects with mobile web services discovery in P2P networks. This subsection discusses the complete discovery process. First the Virtual mobile P2P network is established as discussed in section 5.1. The mobile web service categories and default category groups are then advertised into the network. When a Mobile Host joins the network, it first starts its JXME edge peer features, declares its existence by joining to the super peer. The Mobile Host then declares its services by publishing the respective MSA's in their respective peer groups (according to the categorization), by joining them. When a mobile service requester searches for a service, the request will be sent to the related super peer, which also acts as the JXME proxy peer. The super peer looks through its cache for the matching MSAs. The list of the resulted services is maintained at this proxy peer. The search will then be forwarded to all known edge peers and the super peers and the results are added to the maintained list. If the number of resulted services is small, the searching results could already be accepted and

send back to the mobile web service requester. Otherwise advanced matching/filtering of services is applied on this list, as discussed in section 5.3.4. The filtered result set is then forwarded to the mobile web service client. The process is shown in the following activity diagram shown in figure 5.8 on the facing page.

Delivering the search results to the client is a step by step process. Initially the bare minimum details of the search results like the name and description of the services, expected duration of availability are provided to the client as a scrollable list. The mobile web service client scrolls through this list and selects the interested service. By selecting the service, the client seeks more in-depth knowledge of the service. Remaining details of the particular service like the service description, location URL etc. are then downloaded to the smart phone. The client can accesses the service from the Mobile Host using the details provided in this second step of providing search results. The two step process is adapted in order to reduce the amount of data to be exchanged over the radio link. This improves the scalability of the discovery approach further. The sequence diagram shown in figure 5.9 on the next page, depicts the complete mobile web service, publishing & discovery process.

As discussed earlier, the mobile phone users in the operator network might not use the web services for the development purpose and they could be 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. The applications in this scenario are stored in *Virtual mobile P2P network* and are exchanged using *P2P file sharing mechanism*. The applications are also returned to the client when the results are ordered in the list in first step of delivering the search results. The client can select these applications and can download and install them on smart phones. The installed applications can be used to access the web services from the Mobile Host.

5.4 Mobile Web Service Invocation in JXTA

For achieving the mobile web service discovery in JXTA, the invocation of these services in JXTA is not mandatory. The P2P system can be used as the distributed repository. Once the services are discovered, they can be accessed from internet using the IP feature, as explained in section 5.2 on page 121. The thesis also studied accessing the services directly from the Mobile Host in the JXTA network. This offers other advantages discussed with mobile P2P network, like the elimination of the need for public IP. For achieving the invocation of services in JXTA, the study proposes a *port forwarding model*.

As already discussed, in section 3.5.3, a pipe is to be created for receiving incoming messages over the JXTA network. So the Mobile Host creates a pipe for the mobile web services deployed with it and advertises the pipe into the mobile P2P network by adding the pipe advertisement along with its peer ID to the MSA of the services. Note that as explained in section 5.3.3, MSA has the `<jxta:PipeAdvertisement>` parameter, which can be used to hold the respective pipe advertisement of the service. A potential

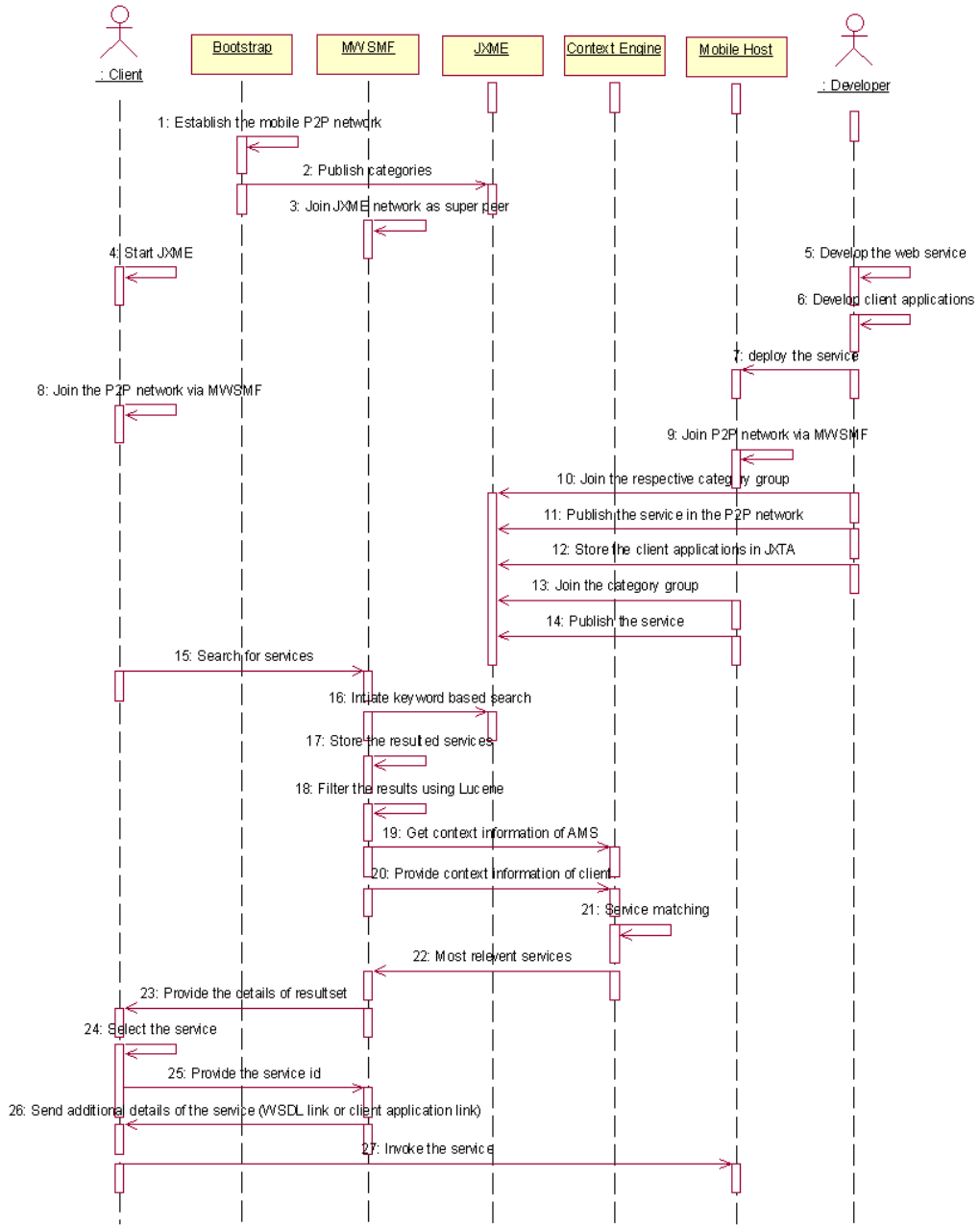


Figure 5.9: Publishing and discovery of mobile web services in JXME networks - Sequence of actions

client searching for a service, after the mobile web service discovery process, gets access to this pipe along with the service description, using the pipe advertisement in the MSA. Using this pipe the client can send the web service request to the Mobile Host. In JXTA the pipes are unidirectional. So the client should create its own pipe for receiving the response from Mobile Host and should advertise along with generating the web service request.

Once the request is initiated, the web service message is received at the Mobile Host on the JXTA default port (9700). But as the Mobile Host follows SOAP over HTTP, it can only receive a web service message over HTTP protocol over port 80. So a JXTA application is to be run on the Mobile Host which receives the web service message over the JXTA, extracts the SOAP contents, and initiates SOAP over HTTP request for the localhost on port 80. The Mobile Host thus receives only the web service requests. The response from the Mobile Host also follows the same root and port forwarding, and the result can be sent back to the client. Alternatively we can modify JXTA access port so that messages can go directly to port 80 at Mobile Host. However this would spoil the main purpose of Mobile Host, where Mobile Host not only serves web service clients from JXTA network but also other web service providers and clients from mobile or non mobile networks like Internet. The approach is currently under study and left for future research directions in this domain.

5.5 Conclusions

This chapter has introduced the concept of providing mobile web services from smart phones in P2P networks. Mobile Host in JXME network offers many advantages in domains like collaborative learning, image sharing, and location based services etc. Not just the enhanced application scope, the JXME network also provides several technical advantages to the Mobile Host like enhanced service discovery and access mechanisms. The discovery issues of mobile web services are discussed and an alternative for mobile web services discovery in P2P networks is suggested using mostly the modules feature of the JXTA. The approach also considered categorizing the services and the advanced features like context aware service discovery. The evaluation of the approach suggested that smart phones are successful in identifying mobile web services in the P2P network, with reasonable performance penalties.

6 Mobile Web Services Mediation Framework

The Mobile Host's QoS and discovery research, discussed in previous chapters, have identified the need for intermediary nodes helping in the successful deployment of smart phone based mobile web service providers in the cellular networks. Based on these requirements a *Mobile Web Services Mediation Framework (MWSMF)* is designed and established as an intermediary between the web service clients and the Mobile Hosts in the mobile network, using the *Enterprise Service Bus (ESB)* technology [section 3.6.1]. The chapter addresses the anticipated deployment scenario for mobile web services in cellular networks and the realization details of the mediation framework.

6.1 Integration Framework for Mobile Web Service Provisioning

In the security analysis of Mobile Host, addressed in section 4.1, it was identified that for securing mobile web service communication not all of the WS-Security specification can be applied to the Mobile Host. The specification was beyond the resource capabilities of today's smart phones. It was also identified that the best means 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 a potential mobile web services client from the Internet can follow full WS-Security standard. This pushes the necessity for some mediation framework as the legitimate intermediary in the mobile web service invocation cycle, transforming the messages to the supported standard. For clarity, if we consider the message encryption scenario, the SOAP messages sent by client can be encrypted with any symmetric encryption algorithm, other than AES-256 like TRIPLEDES, AES-128, AES-192 etc. Since the Mobile Host can not implement the complete WS-Security, the security of the message is to be verified at the intermediary and the message is to be encrypted again using AES-256 before sending the message across the radio link to the Mobile Host, by following the best principles suggested by the security analysis. The mediation framework is also needed to keep track of security keys of both the smart phones and the mobile web service clients that participated in the invocation cycle.

Moreover, verifying security at the mediation framework itself, also improves the

scalability of the Mobile Host, as the Mobile Host now receives only secured and valid requests. All the invalid requests are filtered at the mediation framework itself. So the Mobile Host can process more clients. Similar to the security analysis, the scalability analysis, addressed in section 4.2, also has raised the necessity for an intermediary node. BinXML is not an open standard. Hence not all the messages transmitted over the radio link can be based on this standard. If a client sends an uncompressed message over the mobile network, the transmission is not very efficient, even though the Mobile Host can process such a request. In such a scenario the mediation framework should encode/decode the mobile web service messages to/from XML/BinXML formats in the mobile operator proprietary networks.

Similar to the QoS analysis of the Mobile Host, the *mobile web service discovery*, addressed in section 5.3, also raised the necessity for intermediary nodes, acting as super peers and helping in the JXME publishing and discovery processes, by hosting services like lucene based filtering mechanisms. Apart from this, as discussed in section 5.3.7, 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. So the intermediary should maintain the individual user profiles, personalization settings and context sensitive information of the participating mobile clients and devices for the *context aware service discovery* of mobile web services. Furthermore the study of Semantics-Based Access Control (SBAC) mechanism and its adaptation for mobile web service provisioning domain, addressed in section 4.1.9, demonstrated that the semantic discovery process is heavy, in terms of resource consumption and need to be performed at a standalone intermediary or distributed middleware framework. Based on these requirements for the distributed middleware framework from different domains, the thesis tried to realize a Mobile Web Services Mediation Framework (MWSMF) for smart phones. Before considering the technological and realization details of such an MWSMF, the next section introduces the deployment architecture of such a mediation framework.

6.2 MWSMF Deployment Scenario

The proposed deployment scenario of the MWSMF is shown in figure 6.1 on the next page. The mediation framework is established as an intermediary between the web service clients and the Mobile Hosts in JXTA network. The *virtual P2P network* is to be established in the mobile operator network with one of the node in operator proprietary network, acting as a JXTA super peer, as explained in section 5.1. A super peer has the functionality of both JXTA relay and JXTA 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 thus relay requests to and from JXTA network, to smart phones. Smart phones

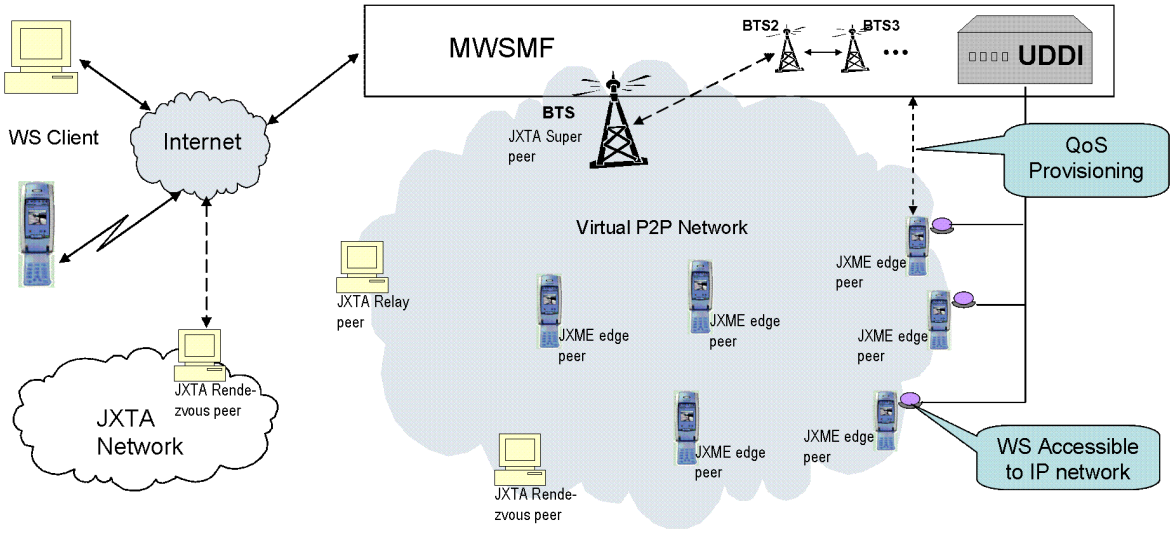


Figure 6.1: The mobile web services mediation framework deployment scenario (Adapted from [Srirama et al., 2006c])

participating in this mobile P2P network use JXME, a light version of JXTA for mobile devices, for connecting to the JXTA network. Standalone systems can also participate in such a network as both rendezvous and relay peers, if the operator network allows such functionality, thus further extending the mobile P2P network [Srirama et al., 2006c].

Once the mobile P2P network is established, the web service clients can discover the services using *mobile P2P discovery* mechanism and can access the deployed services across MWSMF and JXTA network. The mediation framework ensures the QoS of the mobile web service messages and transforms them as and when necessary and routes the messages based on their content to the respective Mobile Hosts. Apart from security and improvements to scalability the QoS provisioning features of MWSMF also includes message persistence, guaranteed delivery, failure handling and transaction support. External web service clients, that does not participate in the mobile P2P network, can also directly access the services deployed on the Mobile Hosts via MWSMF, as long as the web services are published with any public UDDI registry or the registry deployed at the mediation framework and the Mobile Hosts are provided with public IPs. This approach evades the JXME network completely. Thus the mediation framework acts as an external gateway from Internet to the mobile P2P network, as long as the clients in the Internet have the support for JXTA network. The general mobile web services architecture with the Mobile Hosts, shown in figure 3.1 on page 37 of section 3.1.1 on page 36, is still preserved with UDDI support at the mediation framework.

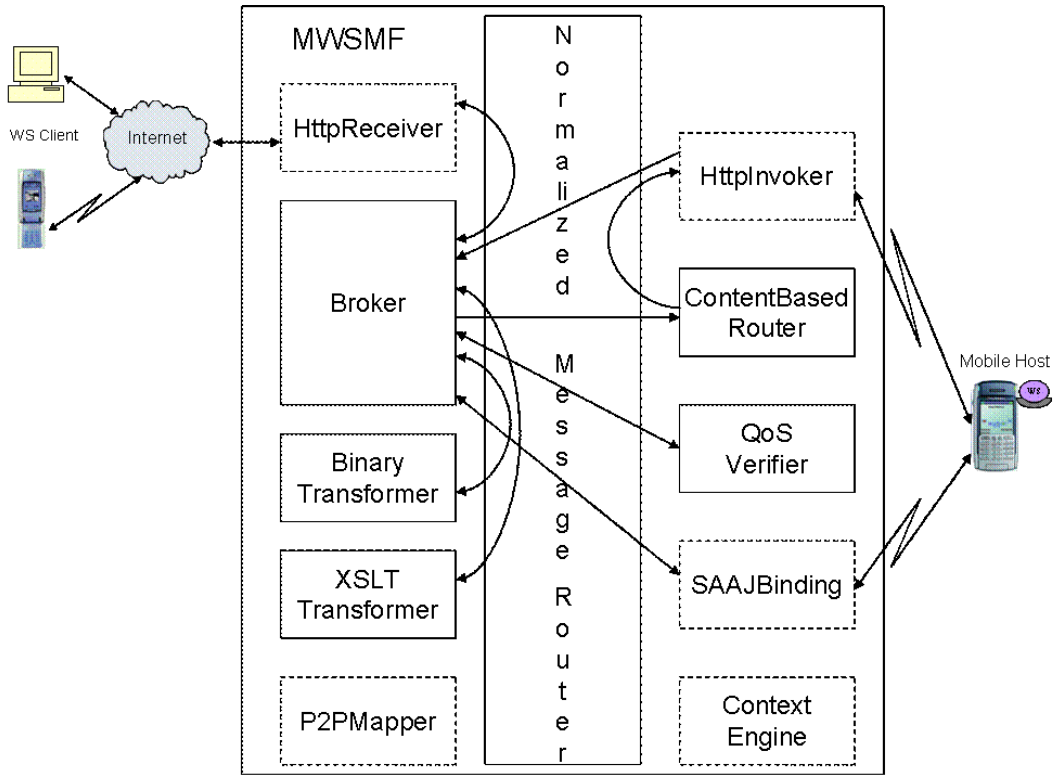


Figure 6.2: Basic components of the MWSMF (Adapted from [Srirama et al., 2007c])

6.3 MWSMF Realization Details

From the MWSMF deployment scenario discussion, addressed in previous section, it can be derived that the MWSMF should have a distributed framework and it should integrate multiple platforms with support for heterogeneous technologies like JXTA, web services etc. ESBs are the recent developments in enterprise integration domain and a standards-based ESB solves the integration problems elevated by the MWSMF. So the study tried to realize the MWSMF based on ESB technology and implemented the middleware framework using JBI based open source ServiceMix ESB. The details of ESB technology, the JBI specification and the features of ServiceMix are explained in sections 3.6.1, 3.6.2, 3.6.3, respectively. Figure 6.2 shows the components of the mediation framework handling different tasks [Srirama et al., 2007c].

ServiceMix by following the JBI architecture supports two types of components - *Service Engine Components* and *Binding Components*. Service engines are components responsible for implementing business logic and they can be service providers/consumers. The service engine components are shown as straight lined rectangles in the figure. The binding components *marshall* and *unmarshall* messages to and from protocol-specific data formats to *normalized messages*. Thus they allow the JBI environment to process

Listing 6.1: Configuration information for the HttpReceiver component

```
<sm:activationSpec componentName="httpReceiver"
                   service="ssn:httpBinding"
                   endpoint="httpReceiver"
                   destinationService="ssn:mwsmfBroker">
  <sm:component>
    <bean class="org.apache.servicemix.components.http.HttpConnector">
          <property name="host" value="localhost"/>
          <property name="port" value="8912"/>
    </bean>
  </sm:component>
</sm:activationSpec>
```

only normalized messages. The binding components are shown as dashed rectangles in the figure. The components are deployed into the framework using spring based xml configuration file. *Spring* is a lightweight container, with wrappers that make it easy to use many different services and frameworks. Lightweight containers accept any Java Bean, instead of specific types of components [Tate and Gehtland, 2005]. The configuration uses the WS-Addressing for routing the messages across the components via the normalized message router. WS-Addressing is a specification of transport-neutral mechanisms that allows web services to communicate addressing information. It essentially consists of two parts: a structure for communicating a reference to a web service endpoint, and a set of Message Addressing Properties, which associate addressing information with a particular message [Box et al., 2004]. The following subsections explain MWSMF's components, their deployment details, and introduce the message flows in the mediation framework across these components under different scenarios.

6.3.1 Components of the MWSMF

HttpReceiver: The HttpReceiver component receives the web service requests (SOAP over HTTP) over a specific port and forwards them to the Broker component via NMR. The component thus acts as the gateway to the mediation framework and as a proxy for the Mobile Hosts. The component is configured into the mediation framework by adding the following xml chunk (shown in listing 6.1) to the configuration file. The `componentName` attribute of the `activationSpec` specifies the name of the component, while `service` indicates the service name of the proxied endpoint, the `endpoint` specifies the endpoint name of the proxied endpoint and the `destinationService` attribute specifies the service name of the target endpoint, here it is the Broker component. The remaining properties of the bean component are specified in the `<sm:component>` element.

The support for this component is deprecated in ServiceMix and ServiceMix ESB now provides a JBI compliant HTTP/SOAP binding component named `servicemix-http`. The component can be used as both a requester and a provider [Apache ServiceMix,

2007]. The component also has the support for WS-Addressing specification. But as most of the message flow analyses of the MWSMF were conducted by considering the HttpReceiver component, here the thesis still goes with the old architecture. As explained already in section 3.6.3, ServiceMix is an open source project under rigorous development and hence the modification to the old components and support for new components are quite obvious and regular. The experience of the thesis with the use of open source tools and softwares is addressed in section 8.1.1 on page 186.

HttpInvoker: The binding component generates a web server request, if the message is a normal HTTP request. The component can also invoke web services by transferring the SOAP messages as HTTP body. The component is developed, while evaluating the mediation framework. As discussed already and similar to HttpReceiver component, the HttpInvoker component can theoretically be replaced by servicemix-http binding component.

Broker: This component serves as a hub for all the communication that happens with other components, in the mediation framework. It receives the client-supplied message from the HttpInvoker component and hosts the main integration logic of the mediation framework. For example, in case of the scalability maintenance, the messages received by Broker are verified for mobile web service/BinXML messages. The component also interfaces with other components and provides the result to the client.

BinaryTransformer: The service engine component transforms the mobile web services messages to and from the BinXML format. If the request message is in the XML format it encodes the message to BinXML format and decodes to XML, if it receives a BinXML message. The component always acts as a provider and it receive the messages from the Broker.

SAAJBinding: The SAAJBinding component from ServiceMix has the support for invoking web services, using SOAP with Attachments for Java (SAAJ) [Jayanti and Hadley, 2005] and Axis via JBI. The component supports only an *InOut message exchange pattern*. The output of the message exchange is replaced by a fault, if the web service invocation is not successful. As already discussed, the functionality can also be achieved with servicemix-http component.

QoSVerifier: The component mainly helps in maintaining the security for the mobile web service messages. The component verifies the security of the received messages and transforms the message to the security level supported by the mobile web services, i.e. the messages are encrypted with AES-256 symmetric key encryption algorithm, and the keys are exchanged with RSA 1024 bit asymmetric key exchange mechanism. The messages are also signed with RSAwithSHA1, as and when necessary. The component uses the XSLTTransformer component for transforming the message to the specified format.

XSLTTransformer: The XSLTTransformer is used to transform mobile web service message into a different format using the XSL documents. The messages can also be transformed to BinXML format using this component. In the current state of implementation, the component is mainly used in security verification of the messages. In the simplest form of security, where no security is provided at the radio level, the component removes

the security headers from the SOAP messages.

ContentBasedRouter: The service engine component, mainly examines the message content, and routes the messages to the corresponding target services, based on data enclosed in the received request messages. The routing can be based on a number of criteria such as existence of fields, specific field values etc. In the mediation framework the component is mainly used to verify the security algorithms of the received messages, so as to decide whether the message needs to be transferred to the QoSVerifier or not, using *XPath* predicates. The component generally acts as provider to the Broker component. If the message has no security support it will be returned back to the broker. The component adapts the `servicemix-eip` component provided by ServiceMix. The servicemix-eip component is a routing container where different routing patterns can be deployed as service unit. Apart from Content-Based Router the component supports routing patterns like Message Filter, XPath Splitter etc.

P2PMapper: The binding component should transform the JXTA messages to/from normalized messages and thus making the invocation of the mobile web services feasible across the JXME network. Presently the mobile web service invocations are only through the HttpInvoker and SAAJBinding components. The component is to be addressed by the future research in this domain.

ContextEngine: The component helps in context aware mobile web service discovery at the mediation framework. The component is also to be addressed by the future research directions in this domain.

6.3.2 Message Flows in MWSMF

The MWSMF developed by the study, tried to realize two usage scenarios where the message flow across the mediation framework is required. They are the *Mobile web service message optimization* and the *Mobile web services security verification* scenarios. The scenarios and the detailed message flows are explained here.

Mobile Web Service Message Optimization

The scenario is required for improving the scalability of the Mobile Host [Srirama et al., 2008a]. The messages received by the mediation framework from external clients are compressed using BinXML encoding, and the binary messages are sent over the radio link. The transformation reduces the size of the message to be exchanged and thus improves the performance of the Mobile Host, as explained in section 4.2. The message flows of the scenario, numbered in figure 6.3 on page 153, are as follows:

1. The HttpReceiver component receives the mobile web service request or the HTTP Request from the client.

2. The HttpReceiver sends the message to the Broker through the NMR

3. The Broker examines the message for web service request and transfers the message to HttpInvoker, if it is a normal HTTP request. The Broker transfers the message to the BinaryTransformer component through the NMR, if the message comprises a mobile web service request.
4. The BinaryTransformer component BinXML encodes the message and transfers the response message back to the Broker component.
5. The Broker component sends the message to the HttpInvoker via the NMR.
6. The HttpInvoker generates the request to the Mobile Host by setting the BinXML data to the body of the HTTP message.
7. The Mobile Host processes the request, as explained in section 4.2.1, and sends the response message back to the HttpInvoker.
8. The HttpInvoker sends the response back to the Broker via the NMR.
9. The Broker transfers the response to the BinaryTransformer through the NMR.
10. The BinaryTransformer decodes the BinXML data to the XML format and transfers the response message back to the Broker component.
11. The Broker returns the response to the HttpReceiver component through the NMR.
12. The HttpReceiver component returns the response back to the client.

Mobile Web Services Security Verification

In this scenario, the security of the messages is verified at the mediation framework and the messages are secured again for sending them over the radio link. For realizing the simplest end-to-end scenario, the study considered no security over the radio link and it was assumed that the security is handled by the operator network. Main reason for this consideration is as there is no standard mechanism to maintain and exchange the asymmetric keys over the radio link for each Mobile Host. Achieving such a key storage mechanism is also quite complex, and it could not be considered in the current thesis. The mechanism is to be addressed by future research directions in this domain. Anyhow the mobile web services security verification scenario won't change as far as the mediation framework design is concerned, with or without the security over the radio link. The message flows of the scenario, numbered in figure 6.4 on page 155, are as follows:

1. The HttpReceiver component receives the mobile web service request or the HTTP Request from the client.
2. The HttpReceiver sends the message to the Broker through the NMR

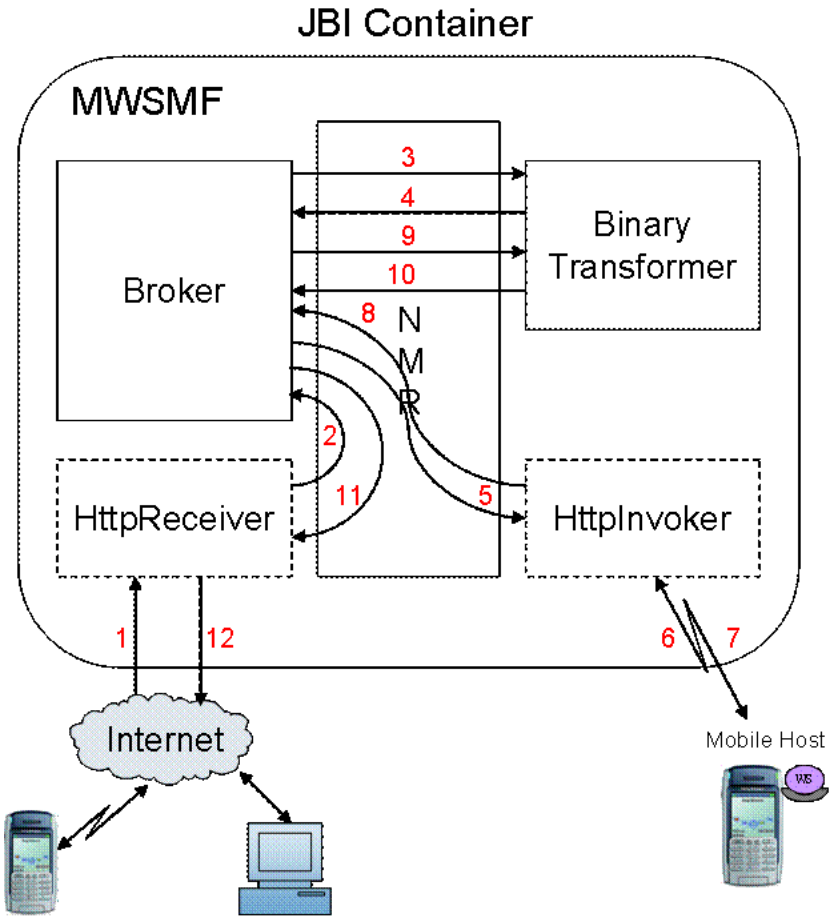


Figure 6.3: Message flows in mobile web service message optimization scenario (Adapted from [Srirama et al., 2008a])

3. The Broker examines the message for web service request and transfers the message to HttpInvoker, if it is a normal HTTP request. The Broker transfers the message to the ContentBasedRouter component through the NMR if the message comprises a mobile web service request.

4. The ContentBasedRouter transfers the message to the QoSVerifier component through the NMR, if the message contains security information. If the message is not secured, it is returned back to the Broker component.

5. The QoSVerifier checks the security of the message and transfers the message to the XSLTTransformer component through the NMR.

6. The XSLTTransformer removes the security headers from the SOAP messages and transfers the response message back to the QoSVerifier component.

7. The QoSVerifier transfers the response messages back to the Broker component.
8. The Broker component sends the message to the SAAJBinding via the NMR.
9. The SAAJBinding component generates the request to the Mobile Host.
10. The Mobile Host processes the request and sends the response message back to the SAAJBinding component.
11. The SAAJBinding sends the response back to the Broker via the NMR.
12. The Broker transfers the response to the QoSVerifier through the NMR.
13. The QoSVerifier incorporates the security parameters to the message and transfers the response message back to the Broker component.
14. The Broker returns the response to the HttpReceiver component through the NMR.
15. The HttpReceiver component returns the response back to the client.

6.4 Supplementary Features of the MWSMF

Apart from acting as the integration framework for mobile web service provisioning, the MWSMF also provides some critical services required in the QoS and discovery maintenance of mobile web services. Some of these features are realized and others are left for the future research in this domain.

1. The mediation framework hosts a UDDI registry. Even though the mobile web services are advertised in mobile P2P network and identifiable with peer ID, smart phones can still posses the public IP feature. The mobile web services can therefore be published with the UDDI registry at the MWSMF. Any external client can search the registry and can access the web services directly. Thus the mediation framework supports the access of web services both across P2P networks and standard web service protocols, acting as an external gateway to the mobile P2P network.
2. MWSMF also supports automatic startup of the Mobile Hosts. Generally hand-held devices have many resource limitations like low computational capacities, limited storage capacities, limited battery power etc. So to conserve these resources, the Mobile Host features of smart phones are to be turned-on only when the provider is prepared to deliver and receives a request from the mobile web service client. The MWSMF identifies the contact details of the phone, when the request is for particular Mobile Host, and sends a Short Message Service (SMS) to the device. A generic program is run on the smart phone that starts the Mobile Host automatically

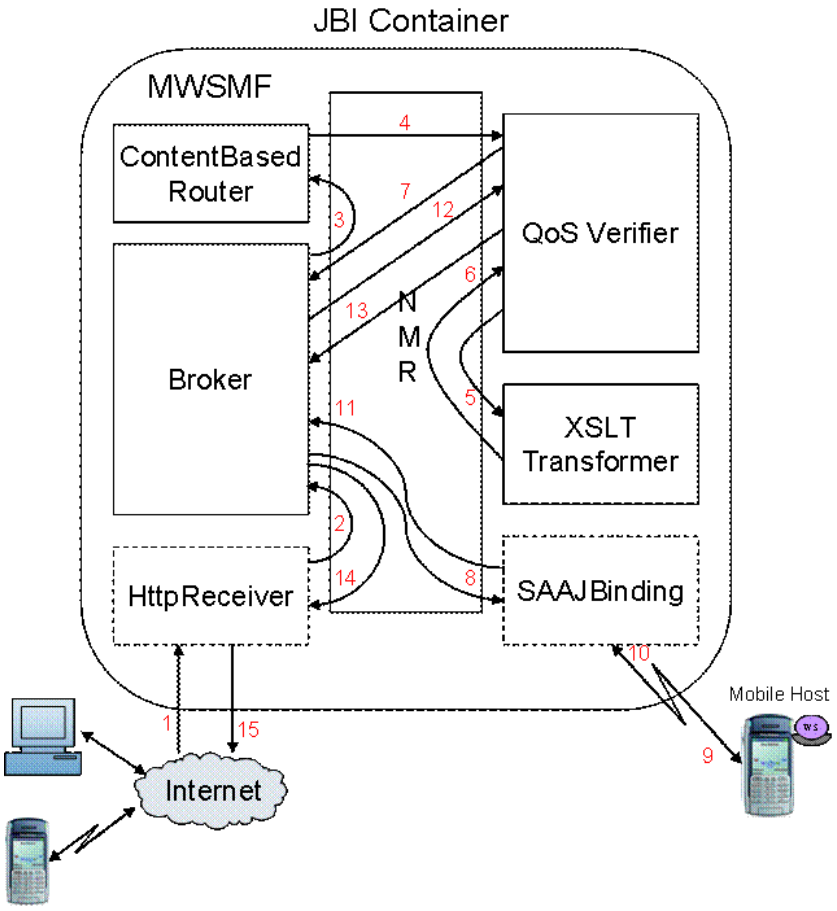


Figure 6.4: Message flows in mobile web services security verification scenario

and activates its services and features, when the SMS message is received.
The application is developed using the *PushRegistry* feature of MIDP 2.0 from Wireless Messaging API (WMA) [Sun Microsystems, 2007i]. WMA is an optional package for J2ME. The PushRegistry support allows a MIDlet to be instantiated automatically to handle in-bound network activity. Combining WMA with the PushRegistry allows a MIDlet to handle SMS messages even if it is not running, when the message is received [Hemphill, 2004]. The SMS messages are sent to the smart phone following specific application protocol. Currently there is support only for the basic features of the Mobile Host like starting the server and authenticating the client, in this regard. The person with the Mobile Host can also opt to turn down this request from client. The scenario is shown as a UML sequence diagram in figure 6.5 on the next page.

3. The MWSMF should store and maintain the asymmetric keys of Mobile Hosts and mobile web service clients, used for analyzing the security of the messages

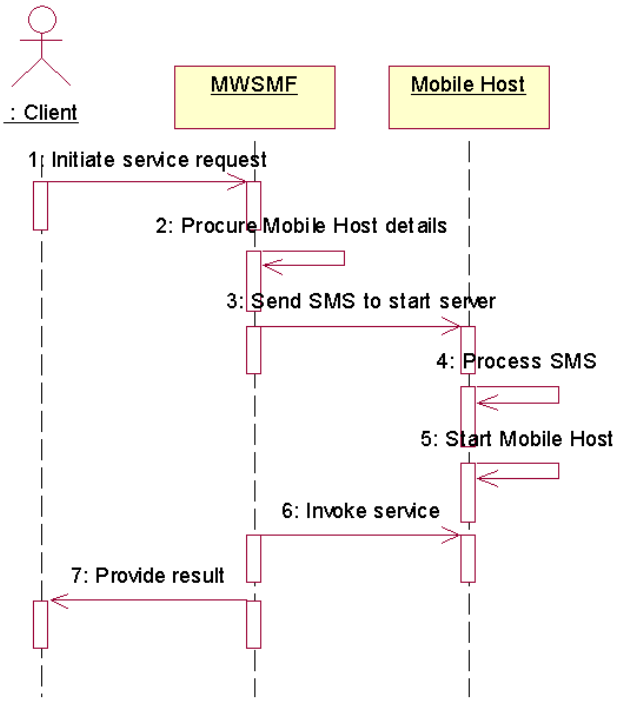


Figure 6.5: Sequence diagram showing automatic startup of the Mobile Hosts

exchanged by participating agents. As already explained the feature is yet to be studied in detail.

4. Similarly the mediation framework should also maintain the context information of Mobile Hosts and deployed services and profiles of the mobile web service clients. This is critical in achieving context aware mobile web service discovery, explained in section 5.3.7.

6.5 Evaluation of the MWSMF

Once the MWSMF was designed and established, the mediation framework was extensively tested for its performance and scalability issues, using load test principles. A huge number of clients were generated for the mediation framework, simulating real-time mobile operator network load. The Expert Rating service considered in the scalability analysis of the Mobile Host, explained in section 4.2.1, is again considered for this performance analysis of the MWSMF. The *mobile web service message optimization scenario*,

explained in section 6.3.2, is mainly evaluated with this analysis.

6.5.1 Test Setup

The ServiceMix based mediation framework is established on a HP Compaq laptop. The laptop has an Intel(R) Pentium(R) M Processor 2.00GHz / 1GB RAM. A Java based server was developed and run on the same laptop on an arbitrary port (4444), mocking the Mobile Host. The server receives the expert rating service request from the client and populates the standard response. The response is then BinXML encoded and the compressed response is sent back to the client, in the HTTP response message format. By considering this simple server, we can eliminate the pure performance delays of the Mobile Host and the transmission delays of the radio link, and thus getting the actual performance analysis of the MWSMF.

For the load generation the study used a Java clone of the ApacheBench load generator from WSO2 ESB [Apache Software Foundation, 2007b; WSO2, 2007]. The load generator can initiate a large number of concurrent web service invocations simulating multiple parallel clients. The command line executable `benchmark.jar` also provides a detailed statistics of the invocations, like the number of concurrent request, successful transactions per second, mean of the client invocation times etc. The `benchmark.jar` and `commons-cli-1.0.jar` are downloaded to a working directory and are used to simulate huge number of concurrent requests. The following sample shows a command that simulates 200 concurrent clients for the expert rating service with each client generating 10 requests for the same service.

```
java -jar benchmark.jar -p ExpertRatingRequest.xml -n 10 -c 200 -H
"SOAPAction: urn:expertRating" -T "text/xml; charset=UTF-8" http://
localhost:8912/soap/Service
```

6.5.2 Test Results

Figure 6.6 on the following page shows the time taken for handling a client request under multiple concurrent requests generated for the mediation framework. The mediation framework was successful in handling up to 110 concurrent requests without any connection refusals. Higher numbers of concurrent requests were also possible, but some of the requests failed as the mediation framework generated '`connection refused IO error`'. The main reason for this connection refusal is as: The ServiceMix transport is based on blocking code which means that the ESB can handle only as many concurrent requests as the number of threads configured in the system [Perera, 2007]. Figure 6.6 on the next page also shows a steady increase in the average time taken for handling a client request with the increase in number of concurrent requests. The figure also shows a sharp decline in the time taken to handle a client after 240 concurrent requests. The decline is because of a large number of failed requests at this concurrency level. More

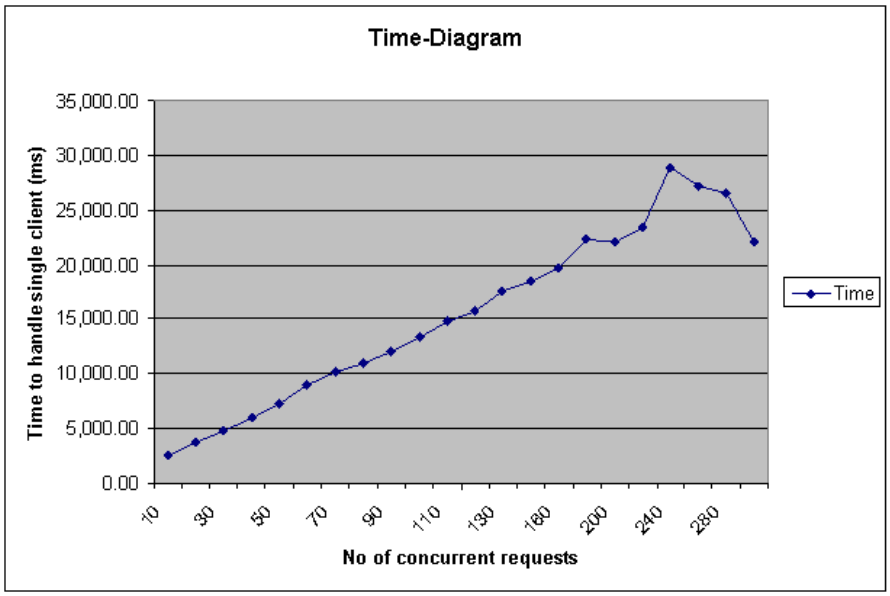


Figure 6.6: Average times taken to handle clients under different concurrency levels, at the mediation framework (Adapted from [Srirama et al., 2007c])

than 300 concurrent requests are not considered as already at this high concurrency level, the number of failed requests is more than 50% of the total requests. The increase in average duration to handle a client is quite normal and the mean duration of handling a single request still remains mostly constant. The mean is calculated considering the performance of the MWSMF over long durations, including parameters like the number of service requests failed. The mean value is in the range 100-150 milliseconds, and it improved slightly with the increase in concurrency levels. This shows the performance of mediation framework is actually improving when there are large numbers of clients to handle.

The results from this analysis show that the mediation framework has reasonable levels of performance and the MWSMF can scale to handling large number of concurrent clients, possible in the deployment scenario explained in section 6.2. This conclusion is also evident from following figure 6.7 on the facing page, which shows that the number of transactions handled by the mediation framework per second almost remains steady (in fact growing) even under such heavy load conditions. The mediation framework is successful in handling 6-8 mobile web service invocations per second, with the mobile web service message optimization scenario. The values can significantly grow, when the deployment scenario is established on reasonable servers, with high resource and performance capabilities.

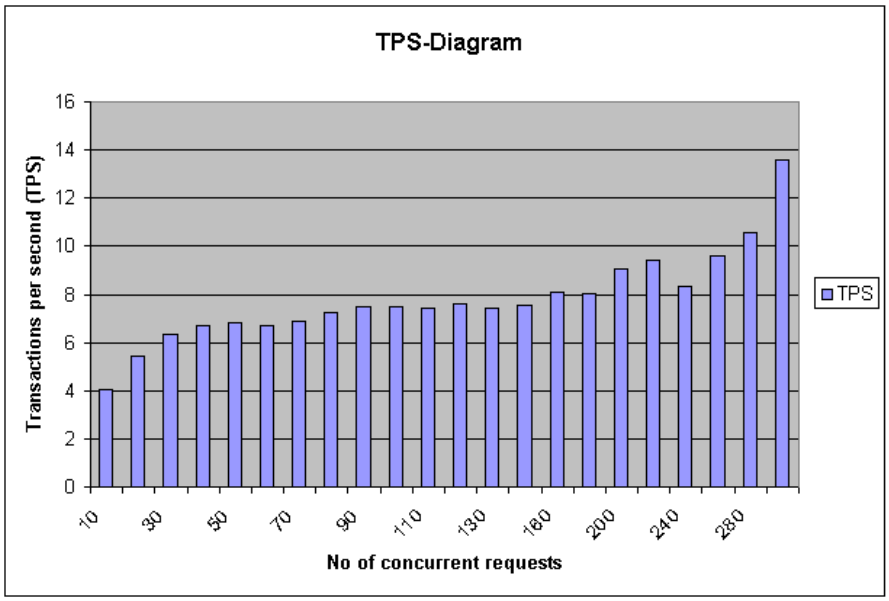


Figure 6.7: The number of transactions handled per second by the MWSMF at different concurrency levels (Adapted from [Srirama et al., 2008a])

6.6 Conclusions

The Mobile Host's QoS and discovery research has identified the need for intermediary nodes helping in the successful deployment of the Mobile Hosts in the cellular networks. So, in the integration analysis of the Mobile Host, an enterprise service bus technology based mobile web services mediation framework was developed, acting as a proxy in mobile web service invocation cycle. The chapter addressed the features, components and realization details of the MWSMF. The regression analysis of the mediation framework conducted with the mobile web service message optimization scenario, clearly showed that the mediation framework has reasonable levels of performance and MWSMF can scale to handling large number of concurrent clients, possible in mobile operator networks.

7 Applications of Mobile Web Service Provisioning

With mobile web service provisioning there is a paradigm shift in mobiles from the role of service consumer to the service provider. This is a step towards practical realization of various computing paradigms such as pervasive computing, ubiquitous computing, ambient computing and context-aware computing. 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. This chapter introduces some of the use cases and application domains of the Mobile Host, and provides the usability evaluation of one such system in m-learning domain showing the adaptability of Mobile Hosts by different communities.

7.1 Mobile Host Application Domains

With Mobile Host, the smart phone can primarily act as a multi-user device without additional manual effort on part of the mobile carrier. Thus the Mobile Host is of significant use in any scenario that needs monitoring and tracking of the mobile carrier's activities. For example, the mobile device can be used to get the location details of an individual, which can be used in scenarios like emergency services, guided tourism etc. Another interesting scenario is with *Remote Patient Tele-Monitoring*, where the Mobile Host gathers patient's data collected from medical sensors attached to the patient's body and delivers this data in a near real-time fashion to the healthcare professionals. Attaching medical sensors and other devices and equipment to the mobiles has become feasible with advancements in technologies like Bluetooth.

Moreover, as a Mobile Host, the mobile terminal can provide access to information like pictures, audios, videos, tags, documents, location details, and other information of individuals in seamless interoperable way. This sort of information can be used in building communities of practice [Wenger, 1998]. In communities of practice, some of the most innovative and valuable information is not made available online, but lies within groups of practice. For example, m-learning communities, groups of specific expedition interests etc. Here the peers can browse through the information, add tags, and give their suggestions or comments.

The Mobile Host was also checked for over the air (OTA) service provisioning capabilities. By this means, services can automatically be downloaded and deployed to the

Mobile Host. The services can be downloaded as individual classes or packaged jar files and can be deployed dynamically to the Mobile Host. The feature was successfully tested with PersonalJava based Mobile Host. The OTA feature makes the Mobile Host provide a platform for smart phones, helping in seamless integration of new mobile applications and services. Examples of such applications can be P2P games, context aware hotel bookings etc. that can be discovered with mobile web service discovery mechanism. The applications or updates can also be automatically installed on all the subscribed Mobile Hosts.

Many interesting applications of Mobile Host are also possible in the area of context-aware computing domain. For example, mobile web service provisioning concept can be used to realize context sources on the mobile device as services and make this context information available to the Context Distribution Frameworks in the Internet. Any mobile application that is attuned to a user's context will leverage knowledge about *who* the user is, *what* the user is doing, *where* the user is, and *which* terminal is in use. Context-awareness helps in reaching the goal of a personalized experience when the user interacts with the mobile services [Pashtan, 2005].

Apart from these specific domains, 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 in a scenario where the mobile is checking for the updates of RSS feeds provided by a server, the mobile has to regularly send requests over the cellular link. The feeds are downloaded by smart phone when an update is detected. This process leaves many unnecessary requests and significantly wastes the bandwidth of the mobile networks. The polling can be eliminated if there is a mechanism to send a message to the mobile phone directly, when there is an update. Thus by providing a mobile web service provider at the smart phone the polling process can be eliminated as the RSS feeds can now be directly sent to all the Mobile Hosts that have subscribed.

The following sections introduce some of the application scenarios of the Mobile Host, demonstrated in mobile community support and collaborative learning domains.

7.2 Collaborative Journalism

An interesting commercial usage scenario of Mobile Host involves the coordination between journalists and their respective organizations. 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 journalists and the content they have gathered. 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.

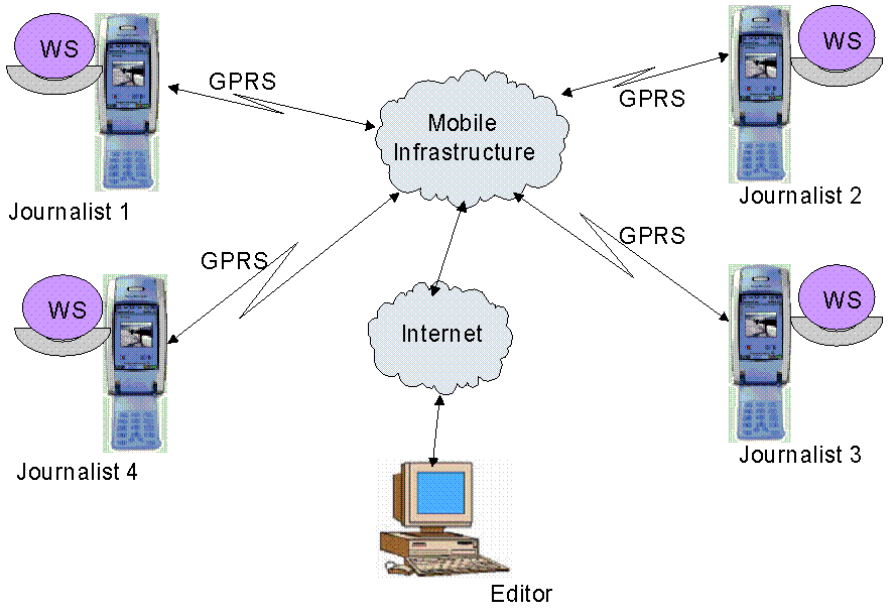


Figure 7.1: The Mobile Host in collaborative journalism scenario (Adapted from [Srirama et al., 2006a])

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 [Srirama et al., 2006a]. The scenario is shown in figure 7.1.

Everyday life usage scenarios driven by regular people are even more interesting. These scenarios of use of mobile web services include citizen journalism, monitoring under crisis management, traffic monitoring, and other. For example of citizen journalism, regular people may create, collect, consume, comment, edit and share news in different media. In average the quality of this information will obviously be lower than professionally prepared news. However, the network effects of participation and the power of *"The Long Tail"*, proved by the second-generation of web based services [O'Reilly, 2005], make this kind of usage scenarios promising.

7.3 Industrial Applications

Another promising industrial commercial area is a business of decentralized network-centric management of powernetworks, which are owned by different businesses. [Naumenko et al., 2007a] have conducted a case study in the domain of distributed power network management. The power networks have geographically distributed complex structures with different equipment and different companies cooperate in order to manage them.

Operators and experts remotely monitor the power networks and prescribe changes.

Field maintenance crews collect information and implement prescribed changes on sites. The communication between different actors may be very important for fault localization, network reconfiguration, network restoration, and other. Moreover, rapid access to the onsite information can highly improve some of the activities and it is especially important for the decision making operators and experts. For example, onsite information about weather conditions, ongoing forest or construction works, or natural phenomena can be used for evaluating existing threats, cause and effects for the power network. This information, which is usually collected by mobile devices, may greatly improve the quality of power-network management. Onsite information may also be used just to extend the operators' view of the power network. Thus, there is a need to integrate existing control systems, tools and application with the mobile devices that are able to provide contextual onsite information. Web services deployed on the mobile devices provide the generic access solution to the information that is collected by the field crews. The scenario is also used to evaluate SBAC mechanism for mobile web services and is explained in detail in section 4.1.9 on page 104.

7.4 MobileHost CoLearn System

As already mentioned Mobile Host offers significant benefits in the collaborative learning domain. This section introduces the MobileHost CoLearn system, developed as a tool helping in learning process. The user evaluation of the system clearly shows the adaptability of Mobile Hosts by different communities.

7.4.1 Collaborative M-learning

Throughout the years there has been a clear evolution from traditional education through distance learning and electronic learning to mobile learning. Traditional education is performed in classrooms, where the teacher presents the learning material to a group of students. The approach benefits from direct contact between teacher and students and thus the ability for immediate feedback. However the approach's disadvantages like the necessity for physical presence of the students in the classroom lead to teaching methodology called *distance learning*, or *d-learning*. Distance learning is characterized with the asynchronous exchange of printed media between teachers and students, sent by the post.

With the advancements in electronic devices and network technologies, another stage of distance learning appeared, i.e. *electronic learning*, or *e-learning*. E-learning can be used either as an enhancement to traditional learning or as a completely electronic-mediated learning environment, which reinforces the learner experience through the use of network-connected electronic devices, such as personal computers. It provides greater adaptability to the learner needs by utilizing multimedia, which can be paused at any time, reversed and replayed as needed.

Subsequently, the rapid development of wireless communication technologies and the advent of mobile devices in everyday life lead to a follow-up of e-learning, called *mobile learning*, or *m-learning*. Papanikolaou et al. define m-learning as a new paradigm, which creates a learning environment, in which "the delivery of educational and/or training programs and/or materials by a mobile device and the dissemination of the material can be done in a synchronous or an asynchronous manner through the use of computer networking technology" [Papanikolaou and Mavromoustakos, 2006]. In general, m-learning can be viewed as any form of studying, performed while the user is interacting with a mobile device. M-learning provides learners the flexibility for not keeping a fixed position during the learning process through the use of portable devices.

Apart from these developments, the view of learning as an isolated individual activity changes to the view of learning as a collaborative process. Learning is no more viewed as being instructor-centered, but a learner-centered process of participating communities of practice, in which learners at various performance levels work together towards achieving a common goal. The various genres of social software, such as photocasting and podcasting, social networking and social tagging, wikis, blogs, etc. and the introduction of m-learning have tremendously changed the Internet, bringing along new opportunities for community building and collaborative learning. In *collaborative m-learning*, locating the right person who can provide us with the right information or the right resources is the best way to find the data that we need.

7.4.2 MobileHost CoLearn System Features

The main functional requirements of the developed MobileHost CoLearn system are described in the use case diagram shown in figure 7.2 on the next page. The system is based on this design and enables learners to:

- Manage their personal and expertise data and make it available to other learners;
- Search for experts in a specific field within a truly collaborative mobile environment; Expand the expert finder search beyond the borders of the social network of the current user by allowing the forwarding of expert finder requests an arbitrary number of times, until a real expert has been found;
- Manage the data of the experts that have been found; Narrow down the list of experts by filtering by name and expertise field with a specific minimum level of expertise;
- Contact experts for further assistance and inquire them about problems faced;
- Manage their own databases of literature resources; organize them into categories of journal articles, conference proceedings, books, URLs, theses etc.; Tag these resources with specific keywords, associated with a three-level scale of relevance

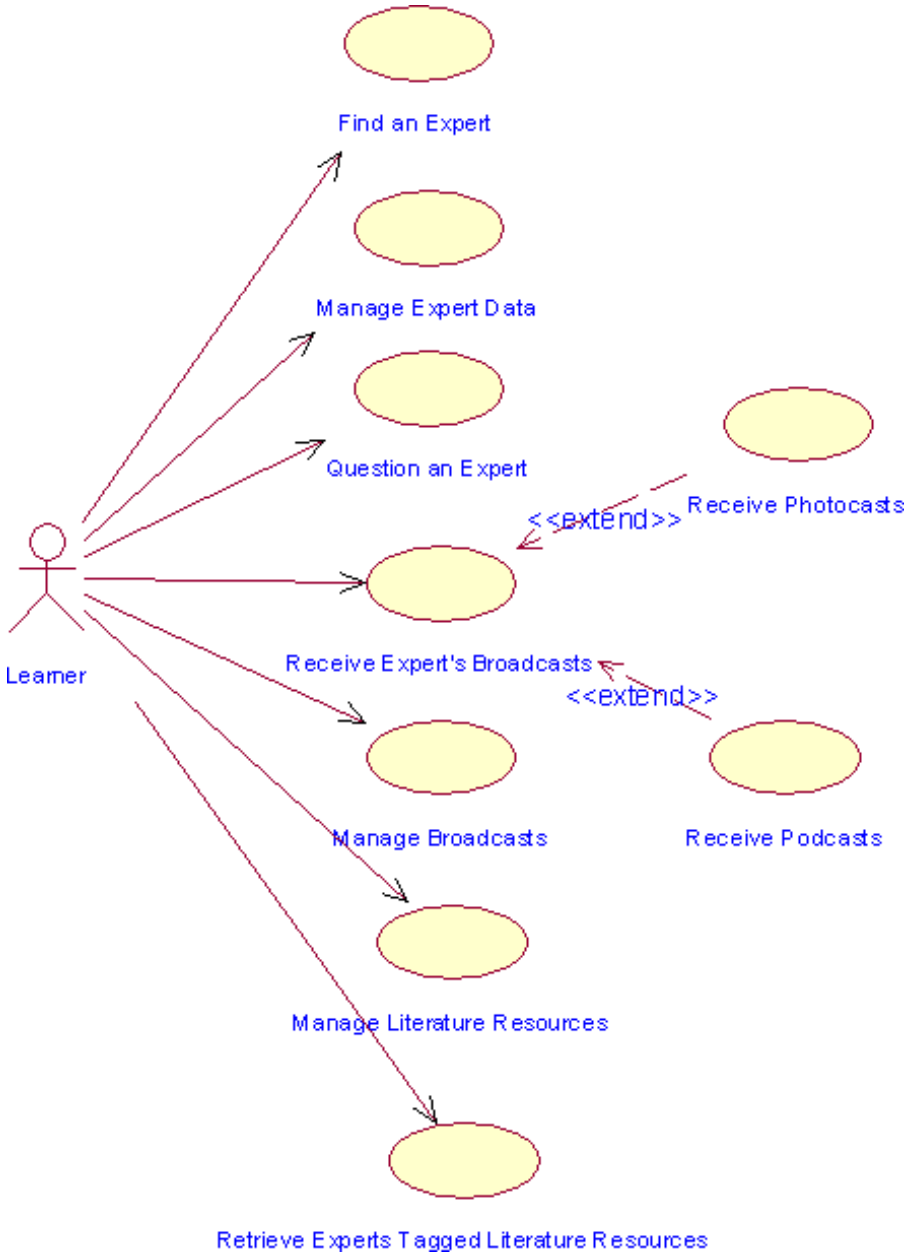


Figure 7.2: Overall MobileHost CoLearn system use case diagram

to the resource (low, medium, high); Narrow down the list of literature by filtering the resources by title, author, tags; Make the literature databases accessible to other learners;

- Retrieve the different types of literature resources of other learners by specifying the tag of interest and its minimum relevance to the resource;
- Manage their photocasting and podcasting channels; Create new photocast episodes by capturing images with the integrated camera device or browsing the file system of the mobile phone, and create new podcast episodes by capturing sound with the integrated audio recording device or browsing the file system of the mobile phone; Manage the metadata of the channels and their episodes; Tag them with specific keywords;
- Subscribe to photocasting and podcasting channels of other learners; Define preferences, such as whether to automatically download the content or just be notified of its availability, and the maximum size of the downloaded content; Unsubscribe from the photocasting and podcasting channels later on;
- Automatically receive new broadcasts as soon as they are available, in case their content complies with the user preferences; Narrow down the list of broadcast episodes by filtering by author and tags;
- Write comments to the received broadcasts; Retrieve the feedback that other learners have left.

7.4.3 MobileHost CoLearn System Modules Hierarchy

The MobileHost CoLearn system consists of three main modules: *Expertise Management Module*, *Expertise Finder Module* and *Expertise Broadcasting Module*. Each of these modules is composed of diverse sub-modules. The module hierarchy and the web services provided by each module are shown in 7.3 on page 169.

The Expertise Management Module contains of three sub-modules: *Expert Management Module*, *Resource Management Module* and *Broadcast Management Module*. Broadcast Management Module further granulated to *Photocast* and *Podcast Management Modules*. The Expert Management Module enables the users to administrate the expert's data, gathered by the Expert Finder Module. Using the Resource Management Module, the learners can organize their literature resources, such as articles, inproceedings, proceedings, books, master and PhD (Doctor of Philosophy) thesis, URLs and unpublished resources, and tag them with keywords with an associated low, medium or high relevance. The Broadcast Management Module is in charge of the different types of broadcasts, created by the current user, such as photocasts and podcasts.

The Expertise Finder Module is composed of three sub-modules: *Expert Finder Module*, *Resource Finder Module* and *Expert Answer Module*. The Expert Finder Module enables

users to search for experts in a specific field via the *Expert Search Web Service*, and receive replies from experts, regarding their level of expertise in the field, via the *Expert Rating Web Service*. A forwarding mechanism for expert finder requests has also been set up, so that a true collaborative environment can be created. After having found an expert, users can utilize the Resource Finder Module to retrieve the expert's literature resources, such as articles, books, etc., which have been tagged with specific keywords, via the *Resource Finder Web Service*. As tagging is quite subjective, a scale of relevance of each tag to each resource has been introduced. The Expert Answer Module gives the possibility for asking specific questions to experts and receiving their answers via the *Problem* and *Expert Answer Web Services*.

The Expertise Broadcasting Module consists of two sub-modules: *Photocasting Module* and *Podcasting Module*. These modules enable learners to subscribe to different types of broadcasts that are delivered by other learners. The types of broadcasts considered are photocasts, dealt within the Photocasting Module, and podcasts (audio & video), dealt within the Podcasting Module. After having retrieved the list of photocasting/podcasting channels, provided by a particular user, via the *Available PhotoCh/PodCh Web Services*, the learners can subscribe to any of them via the *Subscribe PhotoCh/PodCh Web Services* and afterwards automatically receive new episode content, as soon as it is available, via the *Receive Photocast/Podcast Web Services*. As mobile phones are resource limited devices, the learners can set preferences for the received content, such as whether the automatic download of content should be enabled or only a notification should be received, as well as the maximum content size to be downloaded in case the first option is accepted. Users can not only receive learning content, but can also comment on the broadcast episodes and publish their comments on the broadcasting MobileHost CoLearn server via the *Comment Photocast/Podcast Web Services*, as well as retrieve the comments that other learners have left via the *Retrieve PhotoComm/PodComm Web Services*, thus providing a truly collaborative learning environment. The users can unsubscribe from any channel via the *Unsubscribe PhotoCh/PodCh Web Services*.

7.4.4 MobileHost CoLearn System Implementation Details

This subsection explains the MobileHost CoLearn system's class hierarchy. The presentation of the system classes is divided into two batches: the classes that are related to the graphical user interface (GUI), and the classes that are not related to the GUI. The overall class diagram of the MobileHost CoLearn system is presented in 7.4 on page 170. The GUI classes of the MobileHost CoLearn system are presented in the form of a screen flow diagram shown in figures 7.5 and 7.6.

The core class of the system is the `MobCoLearn` class, as shown in 7.4 on page 170. `User` represents a registered user, containing data about the user's name, phone number, e-mail address, other contact details, and copyright. `Server` denotes the Mobile Host server, which listens for incoming requests through a server socket connection on a specific port. `RequestHandler` handles normal HTTP request while `WSHandler` handles web service

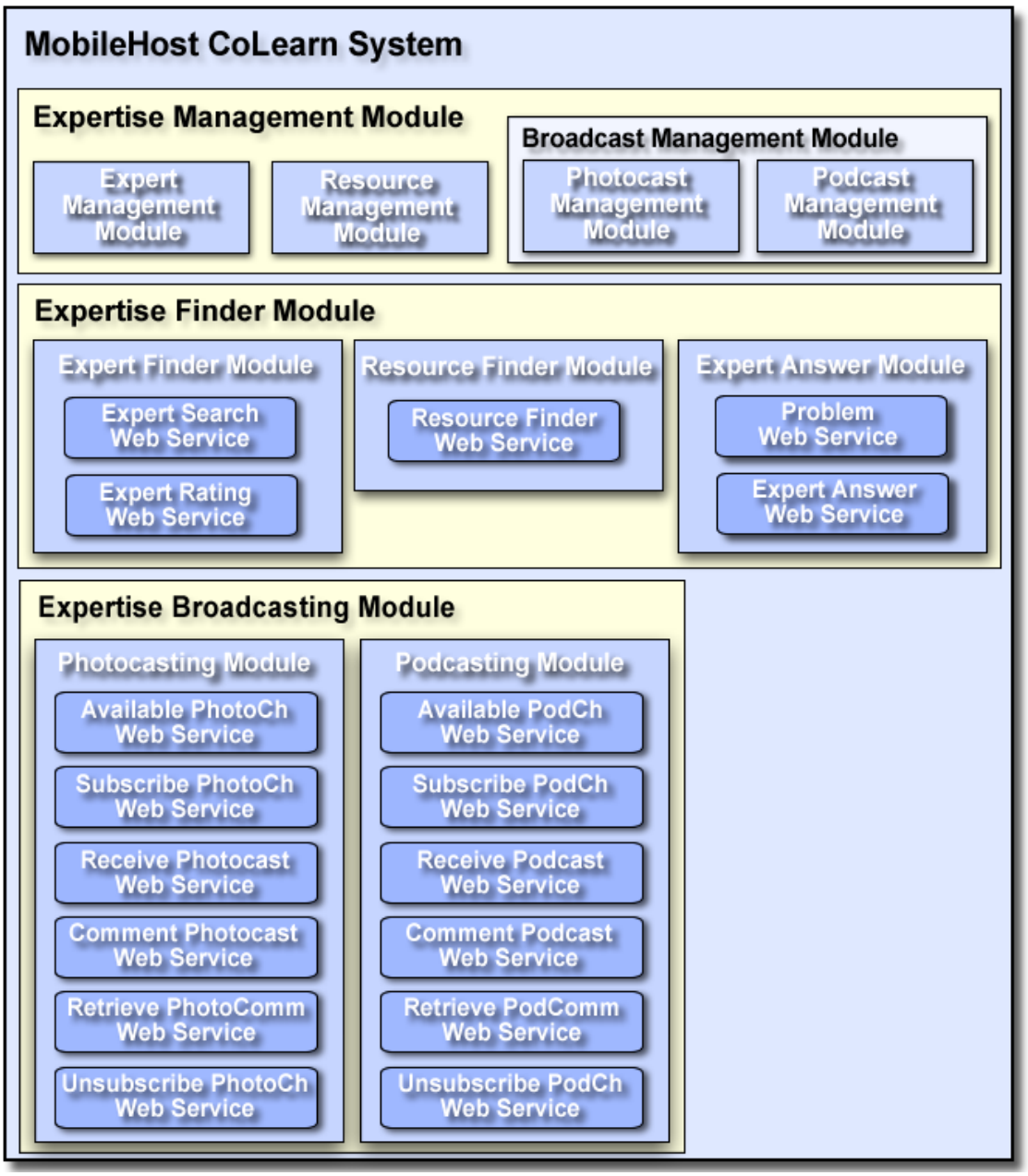


Figure 7.3: Hierarchy of the MobileHost CoLearn system modules (Adapted from [Ivanova, 2007])

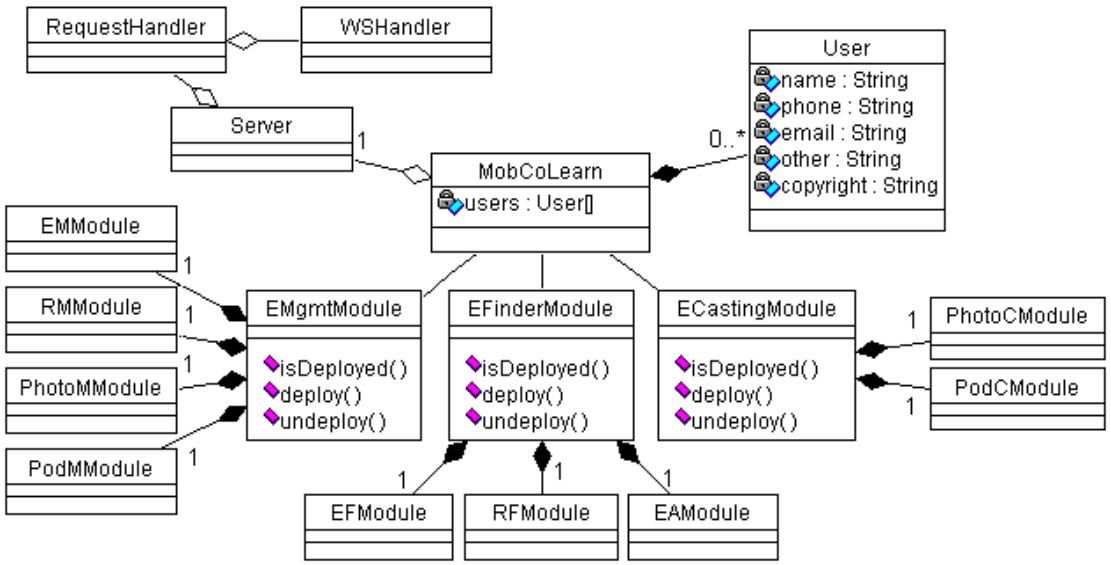


Figure 7.4: Overall MobileHost CoLearn system class diagram (Adapted from [Ivanova, 2007])

requests.

The main modules of the application- Expertise Management, Expertise Finder and Expertise Broadcasting Modules- are represented by classes `EMgmtModule`, `EFinderModule` and `ECastingModule` respectively. The Expertise Management Module embraces its four sub-modules: Expert Management Module (`EMModule`), Resource Management Module (`RMModule`), Photocast Management Module (`PhotoMModule`), Podcast Management Module (`PodMModule`), and provides operations for deploying and undeploying of the module, respectively its sub-modules. The Expertise Finder Module includes its three sub-modules: Expert Finder Module (`EFModule`), Resource Finder Module (`RFModule`), Expert Answer Module (`EAModule`), and provides possibility for deploying and undeploying its sub-modules and their web services. The Broadcasting Module contains its two sub-modules: Photocasting Module (`PhotoCModule`), and Podcasting Module (`PodCModule`), and allows for deploying and undeploying of the sub-modules and their web services. Further details and implementation issues of the sub-modules are omitted here for simplicity and are available at [Ivanova, 2007].

As already mentioned, the GUI classes of the MobileHost CoLearn system are presented in the form of a screen flow diagrams, as shown in figures 7.5 and 7.6. Upon starting the application, the learner should configure the user and server settings, before being able to deal with the web service modules. Screen `SelectUser` displays a list of currently registered users. The learner can directly login with any of the registered users, or first view the details of a user in the `ViewUser` screen, and, if needed, modify them in the `EditUser` screen. If the learner would like to create a new user account, he can do so in the `AddUser` screen. Setting up the server configuration is done in the `ConfigureServer`

screen.

Screen `MainScreen`, as the name suggests, represents the main screen of the MobileHost CoLearn application. It gives a possibility to the user to select if he would like to manage the server status, view or edit the current user and server settings, or work with the currently deployed web service modules. The server can be started and stopped via the `ServerStatus` screen. The current user and server settings can be viewed in the `ViewUserSettings` and `ViewServerSettings` screens respectively by selecting the type of settings in the `Settings` screen, and if needed, these settings can be modified in the `EditUserSettings` and `EditServerSettings` screens.

The list of currently deployed modules, if any, can be viewed in the `Modules` screen. New web service modules can be deployed and currently deployed modules can be undeployed in the `ManageModules` screen. The `ExpertiseMgmtModule`, `ExpertiseFinderModule` and `ExpertiseCastingModule` screens are the main screens of the Expertise Management, Expertise Finder and Expertise Broadcasting Modules. Starting from these screens, the user has access to the respective sub-modules. Screen flows of the sub-modules are available at [Ivanova, 2007].

Notifications regarding the successful or unsuccessful execution of particular actions are displayed in the `AddedUser`, `EditedUser`, `ServerStatusChanged`, `EditedUserSettings`, `EditedServerSettings` and `ManagedModules` alerts.

7.4.5 Expert Finder Module

Since Expert Rating Web Service of the Expert Finder Module is considered for most of the analysis (section 4.2.1 on page 109 & 6.5 on page 156) in this thesis, the module is described here in detail. The Expert Finder Module is used for finding experts in a specific field within a truly collaborative environment. The functionality of the module is accomplished by two web services: the Expert Search Web Service, which should be available on the mobile devices of the users, who will be asked regarding an expert search from an acquaintance- the requestor himself, or a friend of the requestor, or a friend of a friend of the requestor, etc., and the Expert Rating Web Service, which should be available on the mobile device of the requestor, so that he can receive the responses, which will be sent back from experts.

Expert Finder Module Design

The main actors of the Expert Finder Module are the requestor, the forwarder and the responder. A requestor is a user, who in his search for an expert in a specific field creates an expert finder request and sends it to one or more of his acquaintances by invoking the Expert Search Web Service, deployed on their mobile devices. A recipient of the request can be either a forwarder or a responder or a dead-end, a person who is either busy or has no interest and therefore not paid attention to. A forwarder is a user, who serves as an intermediary in the search for experts, by receiving expert finder requests from a

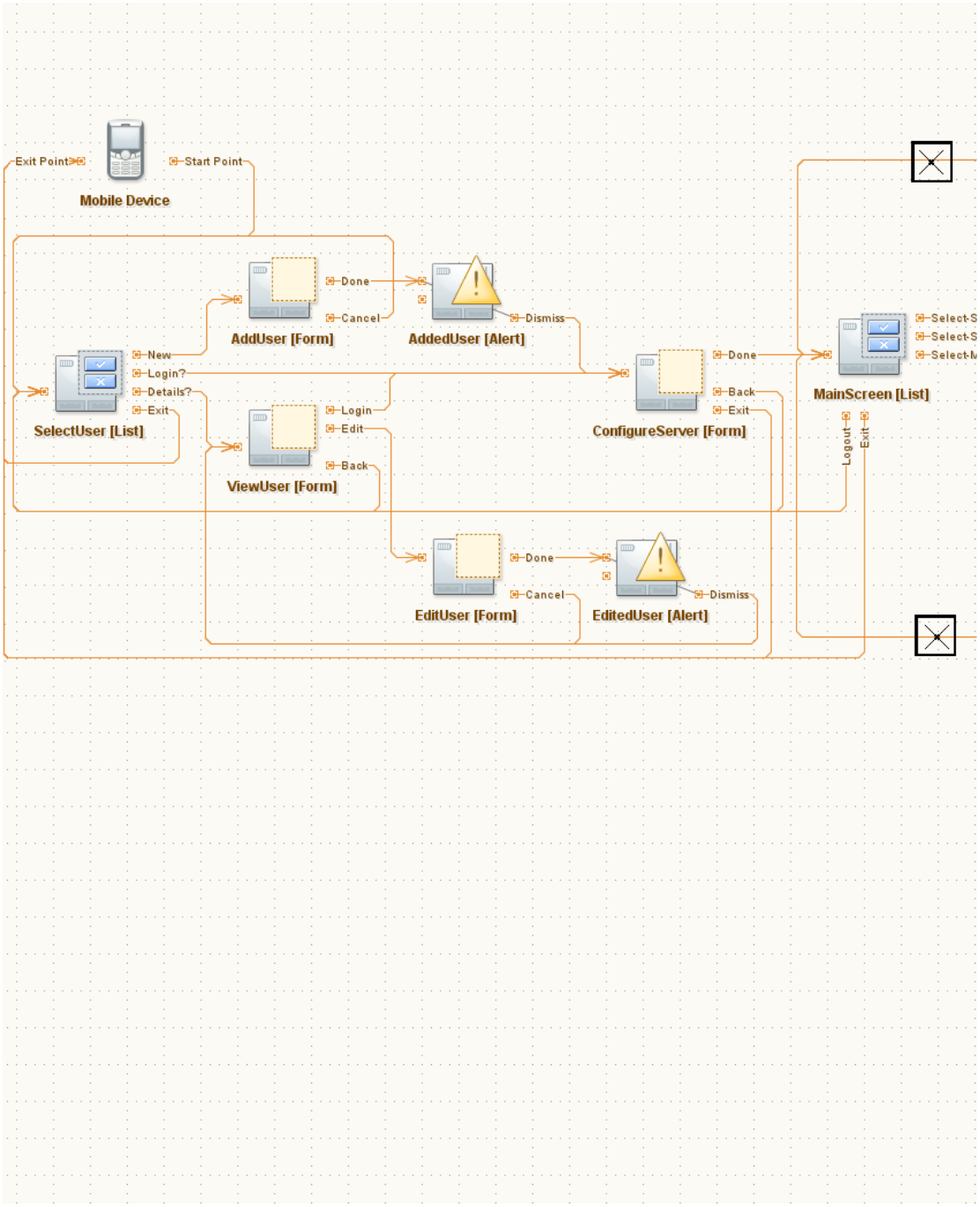


Figure 7.5: Overall screen flow of the MobileHost CoLearn system - Part 1 (Adapted from [Ivanova, 2007])

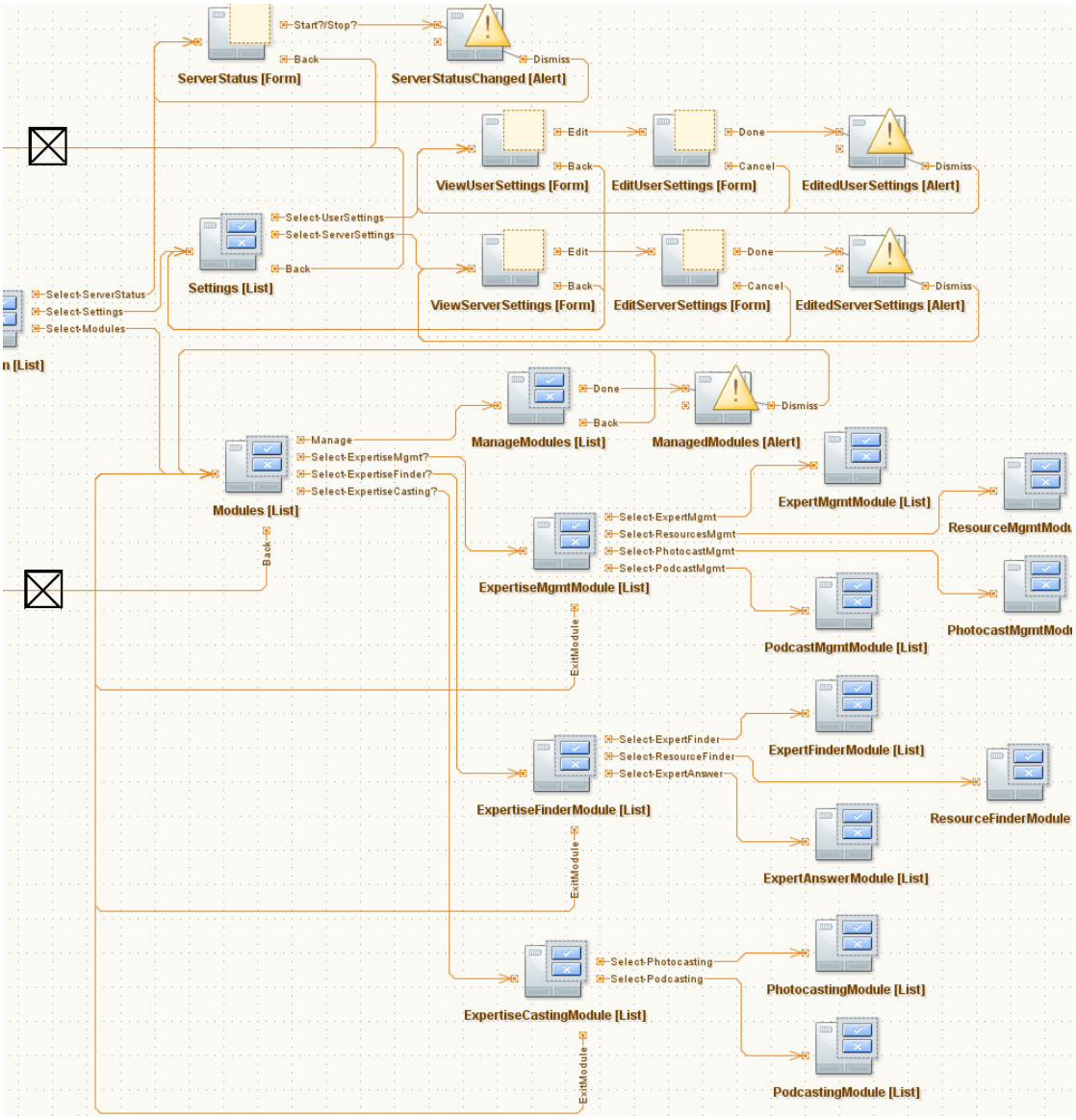


Figure 7.6: Overall screen flow of the MobileHost CoLearn system - Part 2 (Adapted from [Ivanova, 2007])

requestor or from another forwarder, does not assess himself as being an expert in the field, and forwards the request to one or more of his acquaintances by invoking their Expert Search Web Service, installed on his acquaintances' mobile devices. A responder is a user, who receives expert finder requests from a requestor or from a forwarder, assesses himself as being an expert in the given field, and responds to the requestor stating his level of expertise in the field by invoking the Expert Rating Web Service, deployed on the requestor's mobile device.

The use case diagram in 7.7 on the next page depicts the functional requirements of the Expert Finder Module, showing different scenarios of interaction between actors and use cases providing respective functionalities. In his search for an expert, the requestor can create new expert finder requests, import and export requests, view the details of a request, edit or delete a request, send his request to other users- either by inputting their IP address or by selecting them from the contact list of the mobile device- and afterwards view the list of users he has sent his request to, receive back replies from potential experts and afterwards view the replies concerning a request, as well as save a responder as an expert. The forwarder and the responder can receive other users' requests, import and export them, view the details of others' requests, delete them, reply to them and afterwards view their reply, forward the request to other people and afterwards view the list of people they have forwarded the request to.

Services Provided by the Expert Finder Module

As already mentioned, the Expert Search Web Service, in conjunction with the Expert Rating Web Service, composes the web services suite of the Expert Finder Module. The services are expalined here in detail.

Expert Search Web Service

The Expert Search Web Service is responsible for initial sending or forwarding of an expert finder request to other people, and should be provided by the mobile device of the recipient of the request- a potential forwarder or a potential responder. A user can create an expert finder request, stating that he is searching for an expert in a specific field, and send it to one or more of his acquaintances by invoking their Expert Search Web Service, provided by the MobileHost CoLearn applications installed on their mobile devices. As the recipient receives the expert finder request, he can either reply to the requestor, forward the request, or do nothing about it, which is of course not recommended. In case the recipient of the request evaluates himself as being an expert in the field, he can respond back to the requestor, stating his level of expertise, by invoking the Expert Rating Web Service, provided by the Mobile Host of the requestor. In case the recipient of the request does not think that he is an expert in the field, he can forward the request to his own acquaintances, invoking the Expert Search Web Service, provided by their mobile devices.

In a real situation, the expert finding flow is expected to be very complex, in case the requestor sends his request to more than one of his acquaintances, and each of these acquaintances forwards the request to more than one of their own acquaintances, and

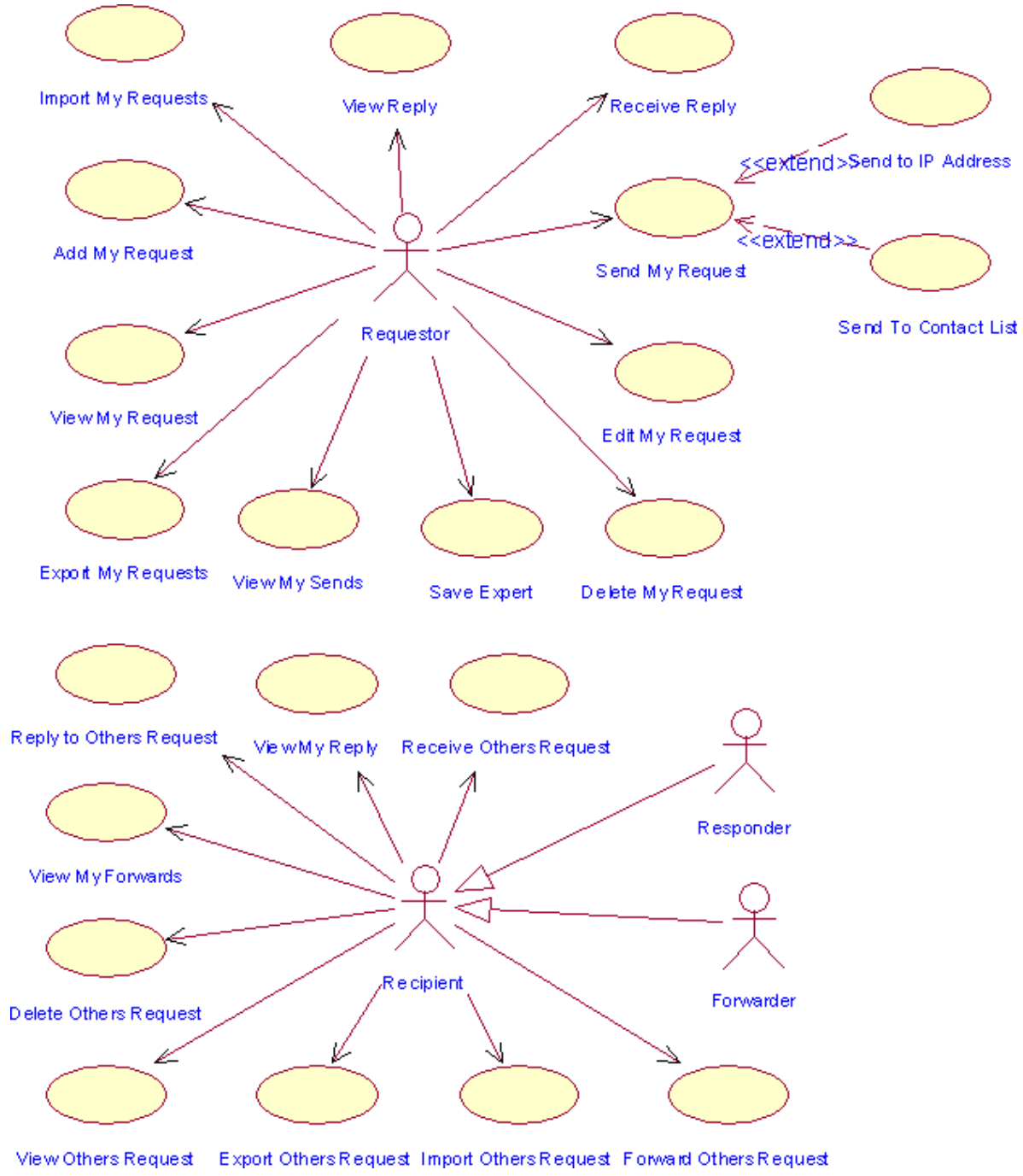


Figure 7.7: Use case diagram of the Expert Finder Module

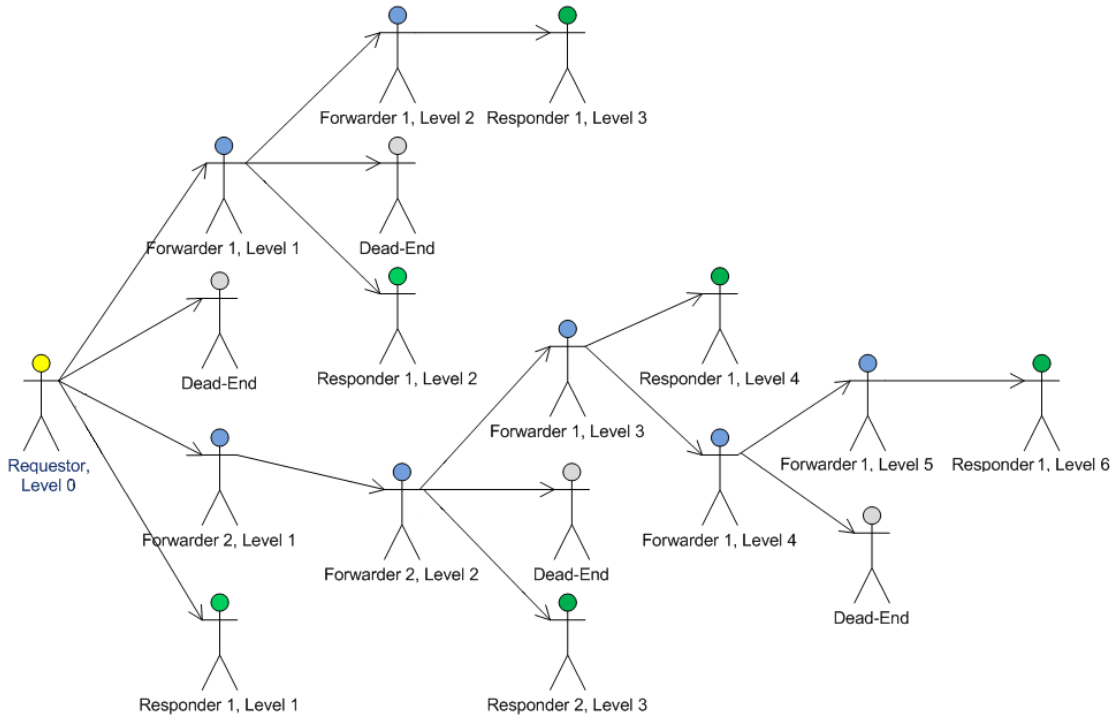


Figure 7.8: Flow of a typical Expert Finder request (Adapted from [Ivanova, 2007])

so on. A sample complex flow, containing various paths through which the different instances of the request travel, is presented in 7.8. In this sample flow, the requestor receives replies from six different people at five different levels, each of whom evaluates himself as being an expert at some level of proficiency. The requestor can then make a subjective decision whom of the experts to contact, either by considering the expert's level of expertise which is returned in the reply, or by looking at the intermediaries through whom the instances of the request have traveled before reaching the experts, and selecting the path consisting of forwarders he knows and trusts more.

The request message format of this web service depends on the path that the request has traveled so far. If this is a request, sent from the requestor at level *0* to users at level *1*, the SOAP request message contains information about the requestor, such as names, phone number, IP address, e-mail address, and other contact data, as well as information about the request itself, such as title, description, start date and expiry date. If this is a request, sent from a forwarder at level k to a user at level $k+1$, the SOAP request message consists of the original request message, complemented with information about the path, which the request has passed so far, i.e. information about each of the forwarders from level *1* to level k, including their names, phone numbers and IP addresses, as well as their comments and the dates of forwarding. The path information is used to avoid loopbacks, and facilitates the requestor in his subjective decision, whom of the experts to contact, in case two or more replies from different experts are received. The structure of this message

is similar to listing 4.4 on page 111 excluding the <*response*> element.

Expert Rating Web Service

If the recipient of the expert finder request assesses himself as being an expert in the given field, he can reply to the requestor by invoking the Expert Rating Web Service, provided by the requestor's Mobile Host. The request message of this service consists of the received Expert Search request message, complemented with his own responder data, and his self assessment of expertise in the field, along with his comment and the date of response.

The request and response messages of the Expert Rating service are provided in listing 4.4 on page 111 and 4.5 on page 112. Altogether, the Expert Rating request message from an expert at level *n* to the requestor at level *0* consists of:

- the original request message, i.e. information about the requestor at level *0*, request title, description, start date and expiry date;
- information about the path, which the request has passed before reaching the expert, i.e. information about each of the forwarders from level *1* to level *n-1*, including their names, phone numbers and IP addresses, as well as their comments and the dates of forwarding;
- the response of the expert at level *n*, including his own responder data, such as names, phone number, IP address, e-mail address and other contact data, as well as the responder's rating for his level of knowledge in the field, his comment and the date of response.

The Expert Rating response message, sent from the requestor at level *0* to an expert at an arbitrary level, contains status information regarding the successful or unsuccessful processing of the request.

Expert Finder Module Implementation Details

The Expert Finder Module is implemented as a sub-module of the Expertise Finder Module. The class diagram of the non-GUI classes of the Expert Finder Module is presented in 7.9 on page 179. The classes, related to parsing of XML files on the file system of the mobile device, permanently storing the data about requests belonging to the current user and requests received from other users, are omitted for simplicity. The presented classes are related to the types of expert search requests, separated into requests of the current user and requests of the other users. The classes are contained within the `expertfinder` package, which is a subpackage of the `efinder` package- the package of the Expertise Finder Module.

`AbstractLearner` is an abstract class, which encapsulates generic information about a learner, such as names, phone number, IP address, e-mail address and other contact data. Classes `Requestor`, `Forwarder` and `Responder` are subclasses of `AbstractLearner`,

representing the different types of users on the request path: these classes do not have any additional attributes, but implement the operation `toXml()`, so that the requestor, forwarder and responder data can be properly exported to an XML file.

`AbstractRequest` is an abstract class, which embraces generic information about a request, be it a current user's request or another user's request, such as title, description, start and expiry dates, requestor. `MyRequest` is a subclass of `AbstractRequest`, representing a request created by the current user, and adding attributes related to the list of people the request has been sent to and the list of replies which have been received back from experts. `OthersRequest` is also a subclass of `AbstractRequest`, representing a request created by another user and received by the current user, which additionally comprises the list of forwards that the request has passed before reaching the current user, the list of forwards that the current user has sent the request to, and the response of the current user, in case it exists. Class `MyForward` represents a forward of the current user, containing the IP address of the recipient and a corresponding comment input by the current user.

`Forward` contains data about a forward from a user who does not assess himself as being an expert in the field to another user, including forward id within the path that the request has passed, forwarder, date and comment from the forwarder to the recipient. `Response` class represents the response of a potential expert, including data about the responder, his own rating for his level of expertise in the field, an optional comment, and the date of response. `Reply` class embraces information about a response from a potential expert and the path that the request has traveled before reaching the expert.

Classes `ExpertSearchService` and `ExpertRatingService` represent the two web services of the Expert Finder Module- the Expert Search Web Service and the Expert Rating Web Service. Class `EFModule` represents the Expert Finder Module and contains the instances of the two web services, the lists of requests created by the current user and the list of requests received from other users.

7.4.6 User Evaluation

Once the MobileHost CoLearn system was developed, the web services of various modules of the system have been extensively tested for their performance. The analysis clearly showed the performance penalties of the system are in line with Mobile Host's services. The technical success of the system has also been verified, making sure that all the functional requirements of the system have been fulfilled. But the main goal of developing the system is to illustrate the adaptability of Mobile Host in different application domains. So the semantic success and effectiveness of the developed system have been evaluated with the help of user testing and a preliminary created questionnaire, bearing in mind the mobility of the learners, and the system's necessity to guideline and support the learners in their new learning situations.

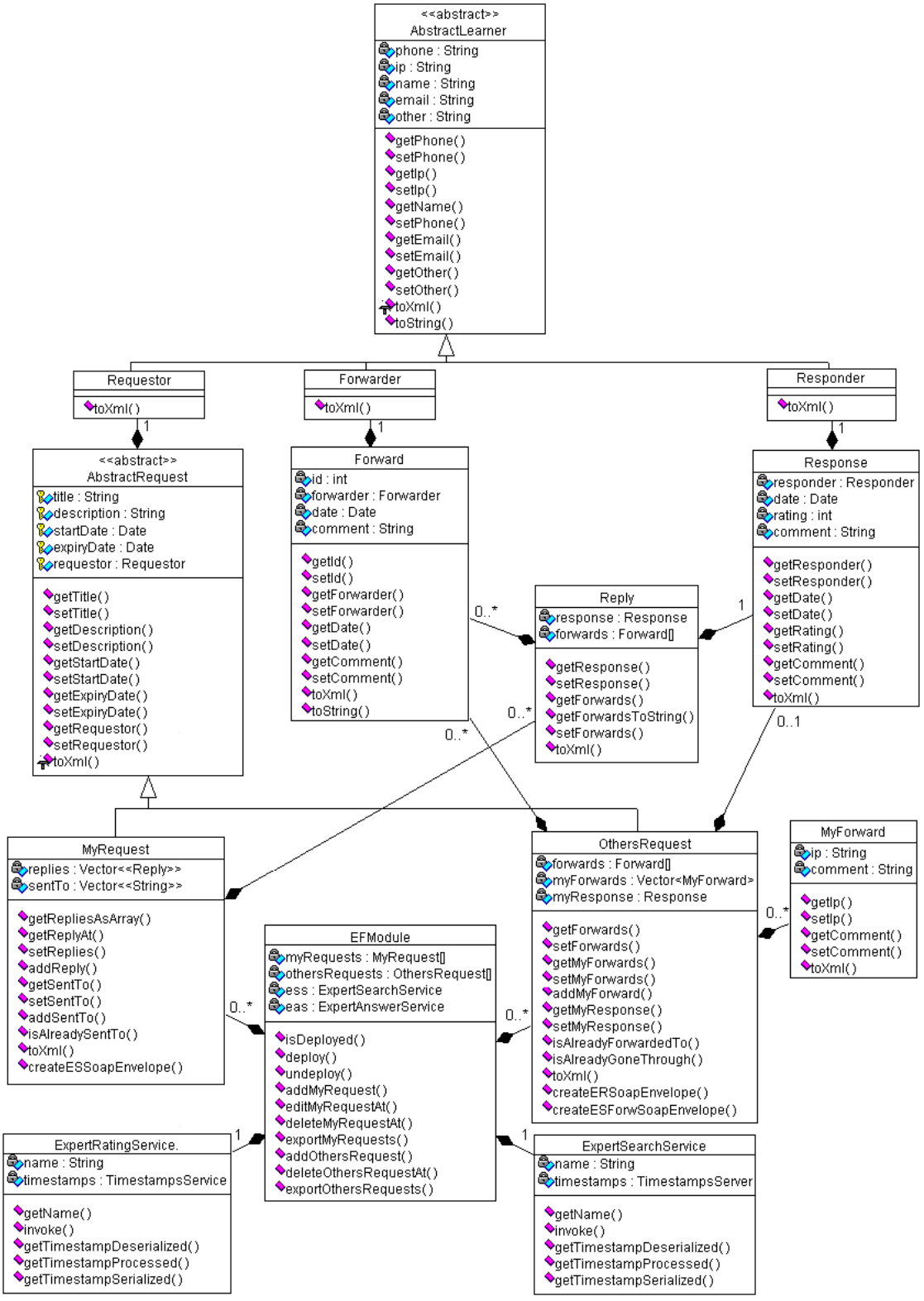


Figure 7.9: Class diagram of the Expert Finder Module (Adapted from [Ivanova, 2007])

Methodology for User Evaluation

The methodology employed in order to test the user experience with the MobileHost CoLearn system as a whole and the user satisfaction with the Expertise Management, Expertise Finder and Expertise Broadcasting Modules, is described here in detail. The users had to perform the following tasks in order to evaluate the functionality of the system and its modules:

1. Read an abbreviated version of the system description.

2. Start the MobileHost CoLearn application, configure the user and server settings, and start the server.

3. Deploy the three main modules: Expertise Management, Expertise Finder and Expertise Broadcasting Modules.

4. Search for an expert and receive expert replies via the Expert Finder Module.

5. Receive expert finder requests from other users via the Expert Finder Module. Forward one of the requests and reply to another request.

6. Save one of the found experts from the Expert Finder Module and manage its personal and expertise data in the Expert Management Module. Create additional expertise fields for an arbitrary expert.

7. Search for tagged literature resources of an arbitrary type via the Resource Finder Module.

8. Save the retrieved literature resources from the Resource Finder Module and manage their data in the Resource Management Module. Create additional related tags for an arbitrary resource.

9. Ask a question to an arbitrary expert via the Expert Answer Module and receive his answer. Reply to a question, asked by another user.

10. Create a photocasting channel via the Photocast Management Module. Create one photocasting episode by selecting an image file from the file system of the mobile device. Create another photocasting episode by capturing an image with the camera of the mobile device.

11. Retrieve the photocast channels of an expert via the Photocasting Module. Subscribe to one of the channels, receive and view a channel episode, comment to an episode, retrieve other's comments, unsubscribe from the channel.

12. Create a podcasting channel via the Podcast Management Module. Create one podcasting episode by selecting an audio file from the file system of the mobile device. Create another podcasting episode by capturing audio from the mobile device.

13. Retrieve the podcast channels of an expert via the Podcasting Module. Subscribe to and unsubscribe from one of the channels.

14. Give feedback about their user experience in the prepared questionnaire.

The user questionnaire is given in Appendix B. The questionnaire has been separated into various sections, covering the personal profile of the user, the test profile, the user's previous experience with mobile applications and collaborative learning applications, the user's overall system evaluation, the evaluation of the each of the system modules, and final remarks. The overall system evaluation section is related to the user experience with the system, without paying attention to the individual modules. The section includes 25 questions, which are a subset of the 50-question database of the Software Usability Measurement Inventory (SUMI)- a rigorously tested and proven method of measuring software quality from the end user's point of view [University College Cork, Ireland, 2007], which has assisted with the detection of usability flaws. The section in the questionnaire actually contains 26 questions, of which question numbered 25 ('Working with this system is a pleasure') is not considered in the usability analysis, as the answers to this question were all 'strongly agree'. The last section of the user questionnaire covers the final remarks and user's suggestions for improving the system.

The user evaluation of the MobileHost CoLearn system has been performed in two sessions. The first session involved 4 users with two of them using mobile devices and the other two users using emulators. 3 users took part in the second session, with one of them using a mobile device and others using emulators. Altogether, 5 females and 2 males tested the system, most of them currently students at the age range 26-31 years. Both sessions durated a little bit more than 2 hours. All the users are very familiar with some of latest mobile softwares like calenders, schedulers etc. and are using them in their everyday life, showing that they are not novices of the domain. But none of the users have used a mobile application for collaborative learning before and none of them was aware of the technology behind the MobileHost CoLearn system.

Overall User Evaluation of the MobileHost CoLearn System

As already mentioned, the statements related to the overall system usability are a subset of the statements, defined by the SUMI inventory. They embrace the user's opinion towards the usability of the system, including measures such as learnability and understandability, the reliability of the system, such as fault tolerance and recoverability, the maintainability of the system, such as stability, the efficiency of the system such as time and resource behavior, and the functionality of the system, such as accuracy and suitability. SUMI is

backed by an extensive reference database embedded in an effective analysis and report generation tool. But, as this tool is highly priced, it has not been used for the analysis of the user feedback. Instead, the System Usability Scale (SUS) [Brooke, 2007], which is also based on a 5-grade scale, has been used for the evaluation of the results.

The result calculated by SUS yields a single number in the range from 0 to 100, representing a composite measure of the overall usability of the system being studied. Each possible answer is assigned a scale position from 1 (strongly disagree) to 5 (strongly agree). The points, contributed by each answer to the total score, depend on whether the statement is positive or negative and on the scale position number of the answer selected by the user. The contribution of each answer to the total score is calculated by the following rule: if the statement is positive, the statement answer contributes *(selected scale position - 1)* points to the total score; if the statement is negative, the statement answer contributes *(5 - selected scale position)* points to the total score. As there are totally 10 questions in the SUS questionnaire, the sum of the points is multiplied by 2.5, to obtain a number in the range from 0 to 100.

The MobileHost CoLearn system evaluation includes 25 questions. So the score need not be multiplied by 2.5 to obtain a final score in the range from 0 to 100. For all the other issues, SUS score calculation has been used. The final scores for the overall satisfaction of each of the 7 users of the MobileHost CoLearn system turned out to be in the range from 78 to 95 points. The individual scores are 78, 82, 85, 89, 91, 95 and 95, in their increasing order. This results in an average user satisfaction of 87.86 points out of 100 points, or approximately 88%.

Regarding the Expert Management Module, the users were quite satisfied with the search possibilities for experts by name and by expertise field with a minimum level of expertise. As an advice for further improvement some of the users suggested to have learner's photos to be available for easy recognition. Regarding the Photocast and Broadcast Management Modules, the users enjoyed the different ways of creating broadcast episodes: creating of photocast episodes by either taking a snapshot with the camera of the mobile phone or browsing the file system for image files, and creating of podcast episodes either by capturing audio from the mobile device of browsing the file system for audio files.

In summary, the users found the GUI of the MobileHost CoLearn system to be simple and intuitive, and were very satisfied with the seamless coupling of different services with the user interface. The users also felt the delays for different activities like getting the services, taking pictures etc., using the system to be reasonable. The users suggested to have a mechanism for automatic discovery of experts, so that if they do not know anyone who might be expert in the field, the system could perform an automatic search.

7.5 Conclusions

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. This is a step towards practical realization of various computing paradigms such as pervasive computing, ubiquitous computing, ambient computing and context-aware computing. The chapter introduced some of the applications studied during the thesis and discussed their realization details. The chapter specifically described MobileHost CoLearn system, developed in the m-learning domain, which presents a truly mobile collaborative learning environment.

In principle Mobile Host offers a large application scope which needs to be explored further by different domains. The research still lacks a killer application. The usability analysis of MobileHost CoLearn system clearly showed the adaptability of Mobile Hosts by learning communities. Similar results are presumed by other communities in further domains.

8 Conclusions and Future Work

The goal of the thesis is to investigate to what degree it is feasible to provide basic mobile web services from smart phones. Chapter 3 showed the technical feasibility of mobile web service provisioning, while chapters 4, 5, 6 showed the services can be provided with reasonable QoS, can be published and discovered with reasonable latencies and the deployment scenario is technically feasible, respectively. Finally, chapter 7, with usability analysis, illustrated that Mobile Host can be adapted by different communities. This chapter concludes the thesis with a summary of the findings and the future research directions. The chapter also provides the experience of the study with using open source tools and softwares.

8.1 Conclusions

The thesis concludes that it is now feasible to deliver basic web services from smart phones due to the advances in wireless devices and mobile communication technologies. The thesis clearly looked at aspects like providing proper quality of service, especially in terms of security and reasonable scalability, discovery of huge number of possible services and integration issues of providing web services from smart phones in operator proprietary networks. The thesis also provided a detailed usability analysis of the Mobile Host, proving the adaptability of Mobile Hosts by different communities.

In the security analysis of the Mobile Host, for ensuring confidentiality and integrity of the exchanged mobile web service messages, the thesis performed a detailed message level security study to adapt some of the best principles of standalone web service like WS-Security for the mobile web services domain. The results of this analysis suggest that the mobile web service messages of reasonable size, approximately 2-5kb, can be secured with web service security standard specifications. The thesis also proposes hardware level solutions for improving these performance latencies. In terms of end-point security ensuring proper authentication and authorization, the basic service-level authentication and user-intervened authorization are realized. The thesis later tried to realize the single sign-on scenario for mobile web services. Further study of this domain lead to the Semantics-Based Access Control (SBAC) mechanism and its adaptation for mobile web service provisioning domain. In this study different deployment scenarios for the SBAC based middleware are proposed and the developed prototype showed applicability of the approach for middleware security guards.

In the scalability analysis of the Mobile Host, the thesis has identified that the Mobile

Host's scalability is inversely proportional to increased transmission delays. In order to reduce the transmission delays, the thesis tried to reduce the size of the mobile web service messages being exchanged, using BinXML compression mechanism, observed to be efficient in mobile web services domain. The study concludes that the performance gain to the Mobile Host with binary compression is quite significant both in terms of improved scalability and battery life.

After the QoS analysis of the Mobile Host, the thesis introduced the concept of providing mobile web services from smart phones in P2P networks. The study concludes that the Mobile Host in JXME network not only enhances the application scope of Mobile Hosts but also provides several technical advantages like enhanced service discovery and access mechanisms. The thesis thus suggests an alternative for mobile web services discovery in P2P networks using the modules feature of the JXTA network. The approach also considered categorizing the services and the advanced features like context aware service discovery. The evaluation of the discovery approach suggested that the smart phones are successful in identifying the services in the P2P network, with reasonable performance penalties for the Mobile Host.

The Mobile Host's QoS and discovery research has identified the need for intermediary nodes helping in the successful deployment of the Mobile Hosts in the cellular networks. So the thesis, in its integration analysis, developed an enterprise service bus technology based mobile web services mediation framework, acting as a proxy in mobile web service invocation cycle. The study addressed the features, components and realization details of the MWSMF. From the regression analysis of the mediation framework, the thesis could conclude that the mediation framework has reasonable levels of performance and the MWSMF can scale to handling large number of concurrent clients, possible in mobile operator networks.

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. This is a step towards practical realization of various computing paradigms such as pervasive computing, ubiquitous computing, ambient computing and context-aware computing. The thesis introduced some of the applications and discussed their realization details. From the usability analysis of the developed MobileHost CoLearn system in m-learning domain, the thesis concludes the adaptability of Mobile Hosts by different communities.

Apart from these conclusions drawn from the thesis, the study also had some observations; with the usage of open source tools for the software development. The details are addressed in the following subsection.

8.1.1 Observations with Open Source Tools Used in the Study

Open source software has become a major interest for the software industry. Open source is a development method for software that harnesses the power of distributed peer review and transparency of process. The promise of open source is better quality, higher reliability, more flexibility, lower cost, and an end to predatory vendor lock-in [Open

Source Initiative, 2008]. As already addressed, in chapters 2 and 3, the thesis mainly considered open standards and open source tools, where ever possible. For example, the study used kSOAP2 for SOAP processing on the mobile devices and ServiceMix tool for realizing the MWSMF.

The main reason for considering these open source tools was the low cost. It is also easy to directly start working with these products, especially for exploring new technologies and standards. Generally open source communities are first to tackle such new problems. For example kXML and kSOAP were one of the first XML and SOAP processors, that can be adapted for resource constrained smart phones. Also with large open source communities, with wide range of technical backgrounds, bugs in open source applications are dealt quickly and efficiently.

The study also observed some problems with using these open source tools. Especially, with such huge community interest, the changes to these products are regular. These changes many times force other modules of the study to be changed regularly, which are directly dependent on these tools. For example in the thesis, the implementation of the Mobile Host had to be adapted from kXML to kXML2 and kSOAP to kSOAP2, with the respective open source upgradations. Most of the times, the backward compatibility of these tools is quite poor. Apart from this, the thesis also observed that most of the performance results discussed are directly dependent on the open source implementations. For example most of the security analysis results produced in the study are dependent on the Bouncy Castle lightweight cryptography API implementation. But then, the peer interest and their reviews of these tools are the proof that most of the results are valid to today's resources.

8.2 Future Research Directions

While the thesis showed that it is feasible to provide basic mobile web services from smart phones, thus integrating personalized mobile services into cellular enterprises and Internet, the study has raised many questions which need to be explored by future work in this domain. In terms of security provisioning for the Mobile Host and achieving end point security, it is still to be studied, how and where to store the security keys and how they can be exchanged. The thesis proposed to store them at the mediation framework, with the MWSMF acting as the identity provider, but still the approach is to be verified. Regarding maintaining the private keys of the users, storing them in the SIM (Subscriber Identity Module) cards of the phones is a prospective solution. As extensions to security analysis, the ideas proposed by the thesis in achieving single sign-on (section 4.1.8 on page 94) and hardware level support for mobile web services security (section 4.1.7), are of significant interest to explore further.

In terms of extensions to the scalability analysis, still it is feasible to find better compression mechanisms than the BinXML. The Binary Format for Metadata (BiM) mentioned in section 3.4.2 on page 54 is a potential target for further exploration. BinXML

is considered as a quick find to illustrate that the compression of the exchanged web service messages over the radio link significantly improves the scalability of the Mobile Host, with the MWSMF and mobile web service message optimization scenario in place. A detailed study of the state of the art in this domain and the study of adaptability of these mechanisms for the Mobile Host need to be verified.

Mobile web service discovery is another domain with full scope for further research. The thesis has identified and demonstrated an alternative for publishing and discovery of mobile web services with the help of P2P networks. The thesis later proposed extensions to this study, considering categorization of services and context-aware service discovery. Regarding the categorization of services, the thesis identified the means of categorizing the mobile web services, but yet the approach is to verified with the UDDI and a comparison is to be drawn. Regarding the context-aware service discovery, the semantic matching of services and introduction of OWL-S based ontologies into this research domain is yet to be verified. Apart from the discovery aspects, invocation of mobile web services in JXME network, using the port forwarding model, proposed in section 5.4 on page 141 is also of significant interest.

Regarding the integration analysis of the mobile web service provisioning, the thesis addressed the features, components and realization details of the MWSMF. As the extensions to the MWSMF, the future research should realize remaining components like the P2PMapper that would join the discovery and integration researches addressed by the thesis, ContextEngine achieving the context aware service discovery at the mediation framework etc. Apart from this, further use cases of the mediation framework are to be found and the scenarios need to realized and integrated, to achieve a reasonable deployment scenario for the mobile web service provisioning. The study might require large scale servers and well designed testbeds.

Regarding the application analysis of the Mobile Host, the thesis has demonstrated several applications, some for proving the technical feasibility and others for proving general purpose user adaptability. Additional creative applications need to be identified, verified and demonstrated, in different application domains. The research still lacks a killer application. The major concern again with any new application is to have a detailed performance analysis that shows the scenario is adaptable to the current generation smart phones.

Apart from these, for performance analysis, the thesis mostly relied on experimental setups rather than theoretical performance models, when and where applicable. With experimental setups and their analysis, the study could observe the actual delays perceivable by the user in cellular networks. Better theoretical approaches like the stochastic simulation models of different parts of the study are of significant interest. But, the simulation results would be as good as the model and can still only be estimates, and proper care is to be taken in designing them.

A Hypertext Transfer Protocol (HTTP)

HTTP is the application layer protocol used to transfer or convey information on intranets and the Web. It is implemented in two communication peers: the client program and the server program, executed on different end systems, and communicating through HTTP messages. The protocol defines the method and the structure of message for the communication.

The HTTP client first initiates a Transmission Control Protocol (TCP) connection to the server port (default port for HTTP is port 80). Once the TCP connection is established, the client sends the HTTP request message to the server through the socket associated with the TCP connection. The HTTP server receives the request through the socket, extracts the path name of the Web page, and retrieves the page from the storage (file system or RAM) of the server, encapsulates the object in an HTTP response message, and sends it to the client using the socket. Once the client receives the message, the TCP connection is closed. The scenario explained here uses the non-persistent TCP connection mode, as the TCP connection closes after processing the client request. A persistent TCP connection is also possible, where the server maintains the connection even after sending the response. Subsequent communication between the same client and server utilizes the already established and still open connection.

The HTTP specification defines two types of message formats, HTTP request and HTTP response messages. The structures of the request and response messages are shown in figure A.1 on the following page and A.2 on the next page respectively. The *Method* field of the HTTP request indicates the method to be performed on the object identified by the *URL*. HTTP defines eight methods indicating the desired action to be performed on the identified resource. Method `GET` means retrieval of the data the URL identifies. Method `HEAD`, which is same as `GET`, returns only the HTTP headers and no document body. Method `POST` submits the data to be processed, to the identified resource. This may result in the creation of a new resource or the updates of existing resources or both. Other methods supported by HTTP include `PUT`, `DELETE`, `TRACE`, `OPTIONS` and `CONNECT`.

HTTP has evolved into multiple, mostly backwards-compatible protocol versions. The Version field of both request and response messages specify the version of HTTP being used by them. The client tells in the beginning of the request, the version it uses, and the server uses the same or earlier version in the response. The supported values are `HTTP/1.0` or `HTTP/1.1` or `HTTP/1.2`.

Similarly, HTTP *header fields* of the HTTP request and HTTP response messages define various characteristics of the data that is requested or the data that has been provided. Some of the prominent header fields are `Content-Length` that specifies the length of the

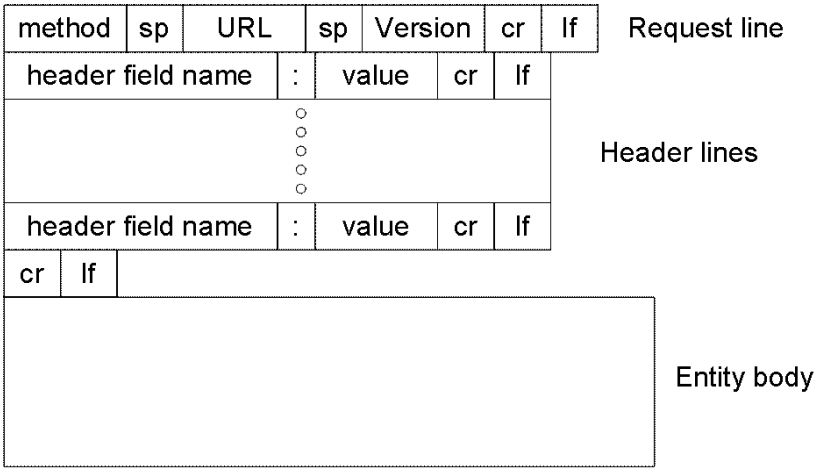


Figure A.1: General format of HTTP request message (Redrawn from [Kurose and Ross, 2001])

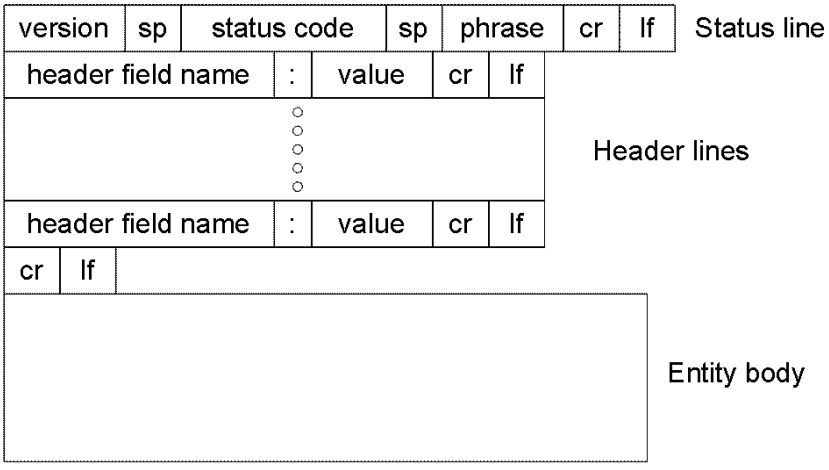


Figure A.2: General format of HTTP response message (Redrawn from [Kurose and Ross, 2001])

content in bytes, `Server` that specifies the name of the server that processed the message and etc. The *status code* of the HTTP response message and the phrase associated with it indicate the result of the HTTP request. Some of the prominent codes include `200 OK` which is standard response for successful HTTP requests, `404 Not Found` which indicates that the requested document does not exist on the server, `505 HTTP Version Not Supported` and etc.

Last but not least critical of the HTTP messages is the *entity body* which contains the requested object itself. The body thus constitutes the main payload of the HTTP messages. The entity body is used with `POST` method and not with `GET` method. The body can be any ASCII (American Standard Code for Information Interchange) message. Thus in SOAP over HTTP transportation of web service messages, the SOAP request is encapsulated into the HTTP request message body and the SOAP response is encapsulated into the body of the HTTP response.

B MobileHost CoLearn System - User Questionnaire

User Evaluation Steps

Dear User,

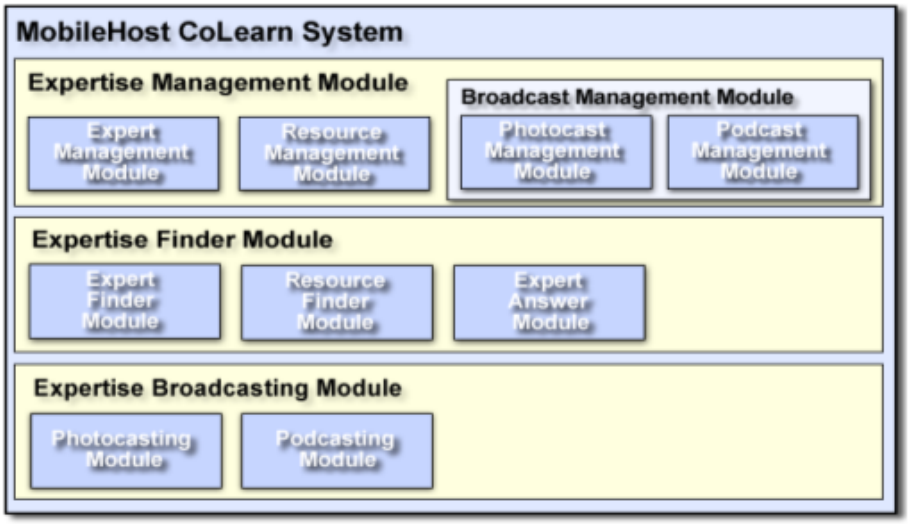
Please perform the following tasks in order to evaluate the functionality of the MobileHost CoLearn system and its modules:

1. Read the abbreviated version of the system description.
2. Start the MobileHost CoLearn application, configure the user and server settings, start the server.
3. Deploy the three main modules: Expertise Management, Expertise Finder and Expertise Boadcasting Modules.
4. Search for an expert and receive expert replies via the Expert Finder Module.
5. Receive expert finder requests from other users via the Expert Finder Module. Forward one of the requests and reply to another request.
6. Save one of the found experts from the Expert Finder Module and manage its personal and expertise data in the Expert Management Module. Create additional expertise fields for an arbitrary expert.
7. Search for tagged literature resources of an arbitrary type via the Resource Finder Module.
8. Save the retrieved literature resources from the Resource Finder Module and manage their data in the Resource Management Module. Create additional related tags for the resources.
9. Ask a question to an arbitrary expert via the Expert Answer Module and receive his answer. Reply to a question, asked by another user.
10. Create a photocasting channel via the Photocast Management Module. Create one photocasting episode by selecting an image file from the file system of the mobile device. Create another photocasting episode by capturing an image with the camera of the mobile device.
11. Retrieve the photocast channels of an expert via the Photocasting Module. Subscribe to one of the channels, receive and view a channel episode, comment to an episode, retrieve other's comments, unsubscribe from the channel.
12. Create a podcasting channel via the Podcast Management Module. Create one podcasting episode by selecting an audio file from the file system of the mobile device. Create another podcasting episode by capturing audio from the mobile device.
13. Retrieve the podcasting channels of an expert via the Podcasting Module. Subscribe to and unsubscribe from one of the podcasting channels.
14. Give us feedback about your user experience with the MobileHost CoLearn system via the questionnaire.

System Description

The MobileHost CoLearn system provides a mobile collaborative learning environment, based on web service provisioning from mobile devices. The system contains three main modules:

- Experise Management Module – Enables the user of the mobile device to manage the data regarding experts in specific fields via the Expert Management Module; manage different types of resource data, such as articles, inproceedings, proceedings, books, URLs, master and PhD theses, and unpublished resources via the Resource Management Module; create and manage photocasts and podcasts in the Photocast and Podcast Management Modules;
- Expertise Finder Module – It allows the user to search for experts in a specific field via the Expert Finder Module, search for resources via the Resource Finder Module, and ask experts for help via the Expert Answer Module;
- Expertise Broadcasting Module – It enables the user to retrieve the available photocast and podcast channels of a particular expert, subscribe to channels, receive channel episodes, and unsubscribe from channels via the Photocasting and Podcasting Modules.



After logging into the MobileHost CoLearn system and defining the user and server settings, the user can select which of the main modules to be deployed: at the time when a main module is deployed, its submodules are deployed as well.

User Questionnaire

Dear User,

Please take a moment to give us your precious feedback about the MobileHost CoLearn System. The questionnaire covers issues regarding your overall system evaluation, as well as your satisfaction with each of the individual MobileHost CoLearn modules. Your opinion is highly appreciated.

User Profile

Name: ……

Occupation / Major: ……

Sex: ❑ male ❑ female

Age: …… years

Brand and type of mobile device, used in everyday life: ……

Mobile phone functionalities, you feel comfortable with:
❑ text input (SMS, E-mail)
❑ calendar and scheduling functionalities
❑ camera recording (pictures, video)
❑ audio recording
❑ picture viewing
❑ video playback
❑ audio playback

Test Profile

Brand and type of mobile device, used during the test: ……

Date of test: ……

Test duration: ……

General Questions

Have you used any mobile applications before? If yes, please list their names.

_____

_____

Have you used any mobile applications for collaborative learning before? If yes, please list their names.

_____

_____

Are you aware of any mobile web service provisioning based applications for collaborative learning? If yes, please list their names.

_____

_____

Overall System Evaluation

Against each statement there are five boxes, representing your response to that statement. You should mark the 1st box if you strongly disagree with the statement, mark the 2nd box if you generally disagree, mark the 3rd box if you are undecided, mark the 4th box if you generally agree, and mark the 5th box if you strongly agree with the statement.

		1	*2*	*3*	*4*	*5*
1.	It takes too much time to learn how to work with the system.	❑	❑	❑	❑	❑
2.	I can understand and act on the information provided by the system.	❑	❑	❑	❑	❑
3.	It is easy to make the system do exactly what you want.	❑	❑	❑	❑	❑
4.	It is easy to see at a glance what the options at each stage are.	❑	❑	❑	❑	❑
5.	There are too many steps required to get something to work.	❑	❑	❑	❑	❑
6.	I sometimes don't know what to do next.	❑	❑	❑	❑	❑
7.	I sometimes wonder if I'm using the right command.	❑	❑	❑	❑	❑
8.	I feel safer if I use only a few familiar commands or operations.	❑	❑	❑	❑	❑
9.	It is relatively easy to move from one part of a task to another.	❑	❑	❑	❑	❑
10.	The system has a very attractive presentation.	❑	❑	❑	❑	❑
11.	The organization of the menus or information lists is quite logical.	❑	❑	❑	❑	❑
12.	The system allows the user to be economic of keystrokes.	❑	❑	❑	❑	❑
13.	The system occasionally behaves in an unexpected way.	❑	❑	❑	❑	❑
14.	Error prevention messages are not adequate.	❑	❑	❑	❑	❑
15.	The system had at some time stopped unexpectedly.	❑	❑	❑	❑	❑
16.	If this system stops, it is not easy to restart it.	❑	❑	❑	❑	❑
17.	The speed of this system is fast enough.	❑	❑	❑	❑	❑

18. The help information provided by the system is not very useful.	❑	❑	❑	❑	❑
19. The instructions and prompts are helpful.	❑	❑	❑	❑	❑
20. The system documentation is very informative.	❑	❑	❑	❑	❑
21. The way the system information is presented is clear and understandable.	❑	❑	❑	❑	❑
22. It is obvious that the user needs have been taken into consideration.	❑	❑	❑	❑	❑
23. Tasks can be performed in a straightforward manner using this system.	❑	❑	❑	❑	❑
24. I feel in command of this system when I am using it.	❑	❑	❑	❑	❑
25. Working with this system is a pleasure.	❑	❑	❑	❑	❑
26. I would recommend this system to my colleagues.	❑	❑	❑	❑	❑

Expert Management Module Evaluation

Did you encounter any problems while:

	yes	no
- viewing experts data	❑	❑
- managing experts data	❑	❑
- viewing expertise fields	❑	❑
- managing expertise fields	❑	❑

Comments regarding technical problems you have faced while using the module:

_____

_____

_____

Do you feel that the Expert Management Module provides the necessary functionalities for managing the data regarding experts, their fields of expertise and their level of expertise? If no, what is missing?

_____

_____

_____

Resource Management Module Evaluation

Did you encounter any problems while:

	yes	no
- viewing resource data	❑	❑
- managing resource data	❑	❑

Comments regarding technical problems you have faced while using the module:

_____

_____

_____

Do you feel that the Resource Management Module provides the necessary functionalities for managing the resource data of articles, inproceedings, proceedings, books, URLs, master and PhD theses, unpublished resources? If no, what is missing?

_____

_____

_____

Photocast Management Module Evaluation

Did you encounter any problems while:

	yes	no
- viewing photocast data	❑	❑
- managing photocast data	❑	❑
- capturing pictures with the camera of the mobile device	❑	❑
- broadcasting your photocasts	❑	❑

Comments regarding technical problems you have faced while using the module:

_____

_____

_____

Do you feel that the Photocast Management Module provides the necessary functionalities for managing your photocasts? If no, what is missing?

_____

_____

_____

Podcast Management Module Evaluation

Did you encounter any problems while:

	yes	no
- viewing podcast data	❑	❑
- managing podcast data	❑	❑
- capturing audio from the mobile device	❑	❑
- broadcasting your podcasts	❑	❑

Comments regarding technical problems you have faced while using the module:

_____

_____

_____

Do you feel that the Podcast Management Module provide the necessary functionalities for managing your podcasts? If no, what is missing?

_____

_____

_____

Expert Finder Module Evaluation

Did you encounter any problems while:

	yes	no
- viewing your own requests	❑	❑
- managing your own requests	❑	❑
- viewing others requests	❑	❑
- sending your own request	❑	❑
- forwarding another user's request	❑	❑
- replying to another user's request	❑	❑

Comments regarding problems you have faced while using the module:

_____

_____

_____

Do you think that the Expert Finder Module provides the needed functionalities for finding experts in specific fields? If no, what is missing?

_____

_____

_____

Do you think that the Expert Finder Module has helped you find an expert?
❑ yes ❑ no ❑ not sure

Are you willing to help other users find an expert?
❑ yes ❑ no ❑ not sure

Do you think that the other users are willing to help you find an expert?
❑ yes ❑ no ❑ not sure

Resource Finder Module Evaluation

Did you encounter any problems while:

	yes	no
- viewing your requests	❑	❑

- managing your requests ❑ ❑
- sending your request and receiving replies ❑ ❑

Comments regarding problems you have faced while using the module:

_____

_____

_____

Do you think that you have been able to define tags and their relevance to resources, matching the tags and relevance defined by the expert?

❑ yes ❑ no ❑ not sure

Do you think that the Resource Finder Module has helped you find useful resources?

❑ yes ❑ no ❑ not sure

Expert Answer Module Evaluation

Did you encounter any problems while:

	yes	no
- viewing your requests	❑	❑
- managing your requests	❑	❑
- viewing another user's requests	❑	❑
- sending your request	❑	❑
- replying to another user's request	❑	❑
- receiving replies from other users	❑	❑

Comments regarding problems you have faced while using the module:

_____

_____

_____

Suggestions for further improvement:

_____

_____

_____

Photocasting Module Evaluation

Did you encounter any problems while:

	yes	no
- subscribing to other users' photocasts	❑	❑
- unsubscribing from other users' photocasts	❑	❑
- receiving another user's photocasts	❑	❑
- viewing other users' photocasts	❑	❑
- writing comments to photocast episodes	❑	❑
- retrieving the comments left by other users	❑	❑

Comments regarding problems you have faced while using the module:

_____
_____
_____

Suggestions for further improvement:

_____
_____
_____

Podcasting Module Evaluation

Did you encounter any problems while:

	yes	no
- subscribing to other users' podcasts	❑	❑
- unsubscribing from other users' podcasts	❑	❑
- receiving another user's podcasts	❑	❑
- viewing other users' podcasts	❑	❑
- writing comments to podcast episodes	❑	❑
- retrieving the comments left by other users	❑	❑

Comments regarding problems you have faced while using the module:

_____
_____
_____

Suggestions for further improvement:

_____
_____
_____

Final Remarks

Do you find that the MobileHost CoLearn system encourages collaborative learning activities?
❑ yes ❑ no ❑ not sure

Do you think that you can learn from fellows and their expertise by using the MobileHost CoLearn system?
❑ yes ❑ no ❑ not sure

Do you feel willing to help others by using the MobileHost CoLearn system?

❑ yes ❑ no ❑ not sure

Are you willing to use the MobileHost CoLearn system in real?

❑ yes ❑ no ❑ not sure

Are there some other scenarios for collaborative learning, which you would like to be available by the MobileHost CoLearn system?

_____

_____

_____

_____

Issues, which you would like to mention, but were not covered by the above questions:

_____

_____

_____

_____

Thank you for taking your time in completing this form!
Your answers will help us shape the future of the MobileHost CoLearn system!

Acronyms

2.5G	Interim Generation
2G	Second Generation
3G	Third Generation
3GPP	Third Generation Partnership Project
4G	Fourth Generation Communications System
AES	Advanced Encryption Standard
AMS	Advanced Matched Services
ANSI	American National Standards Institute
API	Application Programming Interface
ARM	Advanced RISC Machine
ARQ	Automatic Repeat Request
ASCII	American Standard Code for Information Interchange
AWT	Abstract Window Toolkit
BEEP	Block Extensible Exchange Protocol
BiM	Binary Format for Metadata
BPEL	Business Process Execution Language
BTS	Base Transceiver Station
CDC	Connected Device Configuration
CLDC	Connected Limited Device Configuration
CORBA	Common Object Request Broker Architecture
CPU	Central Processing Unit
CSD	Circuit Switched Data
DAML	DARPA Agent Markup Language
DAML-S	DAML Services
DARPA	Defense Advanced Research Projects Agency
DC	Delivery Channel
DCOM	Distributed Component Object Model
DES	Data Encryption Standard
DHCP	Dynamic Host Configuration Protocol
D-Learning	Distance Learning
DNS	Domain Name System
DoD	Department of Defense
DOM	Document Object Model
DOS	Denial-of-Service
DSA	Digital Signature Algorithm
EAI	Enterprise Application Integration
EDA	Event Driven Architecture
EDGE	Enhanced Data rates for GSM Evolution

E-Learning	Electronic Learning
EOL	End-of-life
ESAX	Encoded SAX
ESB	Enterprise Service Bus
eSOAP	Embedded SOAP
FOMA	Freedom of Mobile Multimedia Access
GB	Gigabyte
GHz	Gigahertz
GIS	Geographic Information Systems
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile communications
GUI	Graphical User Interface
HSCSD	High Speed Circuit Switched Data
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	Hypertext Transfer Protocol over Secure Socket Layer
IDE	Integrated Development Environment
IDEA	International Data Encryption Algorithm
ID-FF	Liberty Identity Federation Framework
ID-WSF	Liberty Identity Web Services Framework
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPSec	IP Security
IPv6	Internet Protocol version 6
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
J2EE	Java 2 Platform, Enterprise Edition
J2ME	Java 2 Platform, Micro Edition
J2SE	Java 2 Platform, Standard Edition
JAD	Java Application Descriptor
JAR	Java ARchive
Java ME	Java^{TM} Platform, Micro Edition
JBI	Java Business Integration
JCP	Java Community Process
JDK	Java Development Kit
JMS	Java Message Service
JMX	Java Management Extensions
JSR	Java Specification Request
JTAG	Joint Test Action Group
JVM	Java Virtual Machine
JXME	JXTA for J2ME
JXTA	Juxtapose
KB	Kilobyte
kVM	Kilo Virtual Machine

LA	Liberty Alliance
LAN	Local Area Network
LBS	Location Based Services
LDAP	Lightweight Directory Access Protocol
MB	Megabyte
Mbps	Megabytes per second
MCA	Module Class Advertisement
MCID	Module Class Unique Identifier
MD5	Message-Digest algorithm 5
MEP	Message Exchange Pattern
MHz	Megahertz
MIDlet	Mobile Informational Device Application
MIDP	Mobile Information Device Profile
MIME	Multipurpose Internet Mail Extensions
MIPS	Million instructions per second
M-Learning	Mobile Learning
MMAPI	Mobile Media Application Programming Interface
MMS	Multimedia Messaging Service
MP3	MPEG-1 Audio Layer 3
MPEG	Moving Picture Experts Group
MQ	Message Queue
MSA	Module Specification Advertisement
MSID	Module Specification Unique Identifier
MSP	Mobile Service Platform
MTOM	Message Transmission Optimization Mechanism
MWS	Mobile Web Services
MWSP	Mobile Web Service Provisioning
MWSMF	Mobile Web Services Mediation Framework
NAICS	North American Industry Classification System
NAT	Network Address Translation
NIST	National Institute of Standards and Technology
NMR	Normalized Message Router
NMS	Nomadic Mobile Service
OASIS	Organization for the Advancement of Structured Information Standards
OGC	Open Geospatial Consortium
OMA	Open Mobile Alliance
OS	Operating System
OTA	Over-The-Air
OWL	Web Ontology Language
OWL-S	Web Ontology Language for Services
OWS	OGC Web Services
P2P	Peer to Peer
PAN	Personal Area Networks
PAP	Push Access Protocol
PC	Personal Computer

PDA Personal Digital Assistant
PDAP Personal Digital Assistant Profile
PDF Adobe Portable Document Format
PDP Policy Decision Point
PEP Policy Enforcement Point
PhD Doctor of Philosophy
PIM Personal Information Management
PIMOP Personal Information Management Optional Package
PKI Public Key Infrastructure
PLMN Public Land Mobile Network
PPG Push Proxy Gateway
PPM Prediction by Partial Match
PPP Point-to-Point Protocol
PSTN Public Switched Telephone Network
PXE Preboot Execution Environment
QoS Quality of Service
QRP Query Routing Protocol
RAM Random Access Memory
RDF Resource Description Framework
RDV Rendezvous Peer
RFC Request For Comments
RISC Reduced Instruction Set Computing
RMI Java Remote Method Invocation
ROM Read Only Memory
RPC Remote Procedure Call
RSS Really Simple Syndication
RTT Round-Trip Time
RWTH Rheinisch-Westfälische Technische Hochschule
SAAJ SOAP with Attachments for Java
SAML Security Assertion Markup Language
SAX Simple API for XML
SBAC Semantics-Based Access Control
SDK Software Development Kit
SDRAM Synchronous Dynamic Random Access Memory
SHA Secure Hash Algorithm
SIM Subscriber Identity Module
SIR Session Initiation Request
SMS Short Message Service
SMTP Simple Mail Transfer Protocol
SOA Service Oriented Architecture
SOAP Simple Object Access Protocol
SSL Secure Socket Layer
SSO Single-Sign-On
STL Standard Template Library
SUMI Software Usability Measurement Inventory

SUS	System Usability Scale
SwA	SOAP with Attachments
SWRL	Semantic Web Rule Language
TCP/IP	Transmission Control Protocol/Internet Protocol
TPTP	Eclipse Test & Performance Tools Platform
UDDI	Universal Description, Discovery and Integration
UDP	User Datagram Protocol
UI	User Interface
UIQ	User Interface Quartz
UML	Unified Modeling Language
UMTS	Universal Mobile Telecommunications System
UNSPSC	United Nations Standard Products and Services Code
UPnP	Universal Plug and Play
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
UUID	Universally Unique Identifier
VISR	View based Integration of Web Service Registries
W3C	World Wide Web Consortium
WAP	Wireless Application Protocol
WASP	Web Applications and Services Platform
WBXML	WAP Binary XML
W-CDMA	Wideband Code Division Multiple Access
WLAN	Wireless Local Area Network
WMA	Wireless Messaging API
WS	Web Service
WSA	Web Services Activity
WSA	J2ME Web Services API
WSDL	Web Services Description Language
WS-I	Web Services Interoperability
WSOAP	Wireless SOAP
WTLS	Wireless Transport Layer Security
XACML	Extensible Access Control Markup Language
XML	eXtensible Markup Language
XPATH	XML Path Language
XSD	XML Schema Definition
XSLT	Extensible Stylesheet Language Transformations

Bibliography

3GPP (2007). Third Generation Partnership Project. URL http://www.3gpp.org/. (Cited on pages 16 and 32.)

4GPress (2005). World's First 2.5Gbps Packet Transmission in 4G Field Experiment. 4G Press. URL http://www.4g.co.uk/PR2006/2056.htm. (Cited on pages 16 and 32.)

AgileDelta (2007). Efficient XML: Lightning-Fast Delivery of XML to More Devices in More Locations. URL http://www.agiledelta.com/product_efx.html. (Cited on page 53.)

Aijaz, F. (2006). *Implementation and performance analysis of a UDP binding for SOAP*. Master's thesis, RWTH Aachen University. (Cited on page 31.)

Allan, R. (2004). Systinet WASP. URL http://tyne.dl.ac.uk/ReDRESS/WebServices/webServices_doc/node62.html. (Cited on page 64.)

Alonso, G., Casati, F., Kuno, H., and Machiraju, V. (2004). *Web Services Concepts, Architectures and Applications*. Springer Verlag. (Cited on page 8.)

Anderson, D. P., Cobb, J., Korpela, E., Lebofsky, M., and Werthimer, D. (2002). SETI@home: An experiment in public-resource computing. *Communications of the ACM*, 45(11):56–61. (Cited on page 57.)

Antoniu, G., Cudennec, L., Jan, M., and Duigou, M. (2007). Performance scalability of the JXTA P2P framework. In *IEEE Parallel and Distributed Processing Symposium, 2007*. (Cited on page 135.)

Apache Geronimo (2007). Welcome to Apache Geronimo - Home Page. URL http://geronimo.apache.org/. (Cited on page 73.)

Apache ServiceMix (2007). Apache ServiceMix 3.x Users' Guide. Apache ServiceMix community. URL http://incubator.apache.org/servicemix/users-guide.html. (Cited on pages 72, 73 and 149.)

Apache Software Foundation (2007a). Apache Axis2/Java - Version 1.2. URL http://ws.apache.org/axis2/. (Cited on page 29.)

Apache Software Foundation (2007b). Apache Benchmark. URL http://svn.apache.org/repos/asf/jakarta/httpcomponents/httpcore/trunk/contrib/src/main/java/org/apache/http/contrib/benchmark/. (Cited on page 157.)

Apache Software Foundation (2007c). Apache ServiceMix. URL http://incubator.apache.org/servicemix/home.html. (Cited on page 73.)

Apache Software Foundation (2007d). Web Services - Axis. URL http://ws.apache.org/axis/. (Cited on pages 40 and 81.)

Apte, N., Deutsch, K., and Jain, R. (2005). Wireless SOAP: Optimizations for Mobile Wireless Web Services. In *In the posters of the 14th international conference on World Wide Web*, pages 1178 – 1179. ACM Press. (Cited on page 23.)

ARM (2005). ARM Product Backgrounder. Technical report, ARM. URL http://www.arm.com/miscPDFs/3823.pdf. (Cited on page 15.)

ARM (2007). TrustZone, Security foundation by ARM. URL http://www.arm.com/products/esd/trustzone_home.html. (Cited on page 93.)

ARQ (2007). ARQ - A SPARQL Processor for Jena. URL http://jena.sourceforge.net/ARQ/. (Cited on page 103.)

Atkinson, B., Della-Libera, G., Hada, S., Hondo, M., Hallam-Baker, P., Kaler, C., Klein, J., LaMacchia, B., Leach, P., Manferdelli, J., Maruyama, H., Nadalin, A., Nagaratnam, N., Prafullchandra, H., Shewchuk, J., and Simon, D. (2002). Web Services Security (WS-Security) - Version 1.0. Technical report, IBM Corporation, Microsoft Corporation, VeriSign, Inc. URL http://www.verisign.com/wss/wss.pdf. (Cited on pages 49 and 50.)

Balani, N. (2003a). Deliver Web services to mobile apps. IBM DeveloperWorks. URL http://www-128.ibm.com/developerworks/wireless/edu/wi-dw-wiwsvs-i.html. (Cited on page 18.)

Balani, N. (2003b). Using kXML to access XML files on J2ME devices. IBM DeveloperWorks. URL http://www-128.ibm.com/developerworks/edu/wi-dw-wi-kxml-i.html. (Cited on page 24.)

Ballinger, K., Ehnebuske, D., Ferris, C., Gudgin, M., Karmarkar, A., Liu, C. K., Nottingham, M., and Yendlurif, P. (2007). Basic Profile Version 1.2. Technical report, The Web Services-Interoperability Organization (WS-I). URL http://www.ws-i.org/Profiles/BasicProfile-1.2.html. (Cited on page 14.)

Bar-El, H. (2002). Security Implications of Hardware vs. Software Cryptographic Modules. Technical report, Information Security Analyst. URL http://www.discretix.com/PDF/Security%20Implications%20of%20Hardware%20vs.%20Software%20Cryptographic%20Modules.pdf. (Cited on page 93.)

BEA AquaLogic (2007). BEA AquaLogic Service Bus 2.6: The Enterprise Service Bus for the Agile Business (Product Data Sheet). Technical report, BEA Systems, Inc. URL http://www.bea.com/content/news_events/white_papers/BEA_AquaLogic_Service_Bus_ds.pdf. (Cited on page 69.)

Bellwood, T. (2002). UDDI Version 2.04 API Specification. Technical report, UDDI Committee Specification. URL http://uddi.org/pubs/ProgrammersAPI_v2.htm. (Cited on pages 12 and 130.)

Belov, N., Braude, I., Krandick, W., and Shaffer, J. (2005). Wireless Internet Collaboration System on Smartphones. In *3rd International Workshop on Ubiquitous Mobile Information and Collaboration Systems (UMICS 2005), a CAiSE'05 workshop*. (Cited on page 46.)

Benatallah, B. and Maamar, Z. (2003). Introduction to the special issue on M-services. *IEEE transactions on systems, man, and cybernetics - part a: systems and humans*, 33(6):665–666. (Cited on pages 2 and 18.)

Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The Semantic Web. *Scientific American*, 284(5):34–43. URL http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21. (Cited on page 96.)

Bilorusets, R., Box, D., Cabrera, L. F., Davis, D., Ferguson, D., Ferris, C., Freund, T., Hondo, M. A., Ibbotson, J., Jin, L., Kaler, C., Langworthy, D., Lewis, A., Limprecht, R., Lucco, S., Mullen, D., Nadalin, A., Nottingham, M., Orchard, D., Roots, J., Samdarshi, S., Shewchuk, J., and Storey, T. (2005). Web Services Reliable Messaging Protocol (WS-ReliableMessaging). Technical report, BEA Systems, IBM, Microsoft Corporation Inc. and TIBCO Software Inc. URL http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-rm/ws-reliablemessaging200502.pdf. (Cited on page 30.)

Bloor, R. (2002). PersonalJava Training From SonyEricsson. Wireless Developer Network Symbian DevZone. (Cited on page 81.)

Bolcer, G. A., Gorlick, M., Hitomi, A. S., Kammer, P., Morrow, B., Oreizy, P., and Taylor, R. N. (2000). Peer-to-Peer Architectures and the Magi^{TM} Open-Source Infrastructure. Technical report, Endeavors Technology, Inc. (Cited on page 57.)

Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., and Orchard, D. (2004). Web Services Architecture. Technical report, W3C Working Group Note. URL http://www.w3.org/TR/ws-arch/. (Cited on page 8.)

Borck, J. R. (2005). Enterprise service buses hit the road. *Infoworld*, pages 26–40. URL http://www.infoworld.com/article/05/07/22/30FEesb_1.html. (Cited on page 69.)

Bouncy Castle (2007). Bouncy Castle Crypto APIs. URL http://www.bouncycastle.org/. (Cited on page 82.)

Box, D., Christensen, E., Curbera, F., Ferguson, D., Frey, J., Hadley, M., Kaler, C., Langworthy, D., Leymann, F., Lovering, B., Lucco, S., Millet, S., Mukhi, N., Nottingham, M., Orchard, D., Shewchuk, J., Sindambiwe, E., Storey, T., Weerawarana, S., and Winkler, S. (2004). Web Services Addressing (WS-Addressing). Technical report, W3C Member Submission. URL http://www.w3.org/Submission/2004/SUBM-ws-addressing-20040810/. (Cited on page 149.)

Boyer, J. (2001). Canonical XML, Version 1.0. Technical report, W3C Recommendation. URL http://www.w3.org/TR/xml-c14n. (Cited on page 53.)

Brisaboa, N. R., Luaces, M. R., Parama, J. R., and Viqueira, J. R. (2007). Managing a Geographic Database from Mobile Devices Through OGC Web Services. In *International Workshop on DataBase Management and Application over Networks (DBMAN 2007)*. (Cited on page 19.)

Brooke, J. (2007). SUS - a quick and dirty usability scale. Redhatch Consulting Ltd. URL http://www.usabilitynet.org/trump/documents/Suschapt.doc. (Cited on page 182.)

Brown, N. and Kindel, C. (1998). Distributed Component Object Model Protocol – DCOM/1.0,. Technical report, Microsoft Corporation. URL http://samba.osmirror.nl/samba/ftp/specs/draft-brown-dcom-v1-spec-03.txt. (Cited on page 7.)

Burbeck, S. (2000). The Tao of e-business services - The evolution of Web applications into service-oriented components with Web services. IBM DeveloperWorks. URL http://www-128.ibm.com/developerworks/webservices/library/ws-tao/. (Cited on page 7.)

Cantor, S., Hirsch, F., Kemp, J., Philpott, R., and Maler, E. (2005a). Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0. Technical report, OASIS Standard. URL http://docs.oasis-open.org/security/saml/v2.0/saml-bindings-2.0-os.pdf. (Cited on page 52.)

Cantor, S., Hodges, J., Kemp, J., and Thompson, P. (2007). Liberty ID-FF Architecture Overview, Version 1.2. Technical report, Liberty Alliance Project. URL http://www.projectliberty.org/liberty/content/download/1990/13884/file/liberty-idff-arch-overview-v1.2.pdf. (Cited on page 94.)

Cantor, S., Kemp, J., Philpott, R., and Maler, E. (2005b). Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. Technical report, OASIS Standard. URL http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf. (Cited on pages 49 and 51.)

Capone, A. and Musumeci, L. (1997). Performance Evaluations Of High Speed Circuit Data Services In GSM Systems. In *Proc. IEEE Vehicular Tech. Conf. (VTC '97)*. (Cited on page 31.)

Carlsson, B. and Gustavsson, R. (2001). The Rise and Fall of Napster - An Evolutionary Approach. In *Proceedings of the 6th International Computer Science Conference on Active Media Technology*, pages 347 – 354. Springer-Verlag. (Cited on page 57.)

Chappell, D. A. (2004). *Enterprise Service Bus*. O'Reilly Media, Inc. (Cited on page 67.)

Chatti, M. A., Srirama, S., Kensche, D., and Cao, Y. (2006). Mobile Web Services for Collaborative Learning. In *Fourth IEEE International Workshop on Wireless, Mobile and Ubiquitous Technology in Education - (WMUTE'06)*, pages 129–133. (Cited on page 45.)

Cheney, J. (2001). Compressing XML with Multiplexed Hierarchical PPM Models. In *In Proceedings of IEEE Data Compression Conference*, pages 163–172. (Cited on page 54.)

Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S. (2001). Web Services Description Language (WSDL) 1.1. Technical report, W3C Note. URL http://www.w3.org/TR/wsdl. (Cited on page 10.)

CIA (2006). Smartphones to Outsell PDAs by 5:1 in 2006. Computer Industry Almanac Inc. URL http://www.c-i-a.com/pr0306.htm. (Cited on page 14.)

Clarke, I., Miller, S. G., W.Hong, T., Sandberg, O., and Wiley, B. (2002). Protecting Free Expression Online with Freenet. *IEEE Internet Computing*, 6(1):40–49. (Cited on page 57.)

Cleary, J. G. and Teahan, W. J. (1997). Unbounded Length Contexts for PPM. *The Computer Journal*, 40(2/3):67–75. (Cited on page 54.)

Contreras, P., Zervas, D., and Murtagh, F. (2005). Web Services in Natural Language: Towards an Integration of Web Service and Semantic Web Standards in the JXTA Peer to Peer Environment. URL `http://thames.cs.rhul.ac.uk/~wstalk/papers/rhulpapers/orchestration.pdf`. (Cited on page 65.)

Cunnings, R., Fell, S., and Kulchenko, P. (2001). SMTP Transport Binding for SOAP 1.1. URL `http://www.pocketsoap.com/specs/smtpbinding/`. (Cited on page 30.)

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., and Weerawarana, S. (2002). Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. *IEEE Internet Computing*, 6(2):86–93. (Cited on page 9.)

Cutting, D. (2007). Apache Lucene. URL `http://lucene.apache.org/`. (Cited on page 65.)

Dao, T. (2005). UDDI Explorer: Tool for searching web services. The Code Project. (Cited on page 129.)

Davis, D. and Parashar, M. (2002). Latency Performance of SOAP Implementations. In *IEEE Cluster Computing and The Grid 2002*. (Cited on page 52.)

Dey, A. K. (2000). *Providing Architectural Support for Building Context-Aware Applications*. Ph.D. thesis, Georgia Institute of Technology. (Cited on page 139.)

Dierks, T. and Allen, C. (1999). The TLS Protocol Version 1.0, RFC 2246. Technical report, IETF. URL `http://www.ietf.org/rfc/rfc2246.txt`. (Cited on page 49.)

Discretix (2007). Core Technology Overview. URL `http://www.discretix.com/tech.shtml`. (Cited on page 93.)

Dustdar, S. and Treiber, M. (2006). Integration of transient Web services into a virtual peer to peer Web service registry. *Distributed and Parallel Databases*, 20:91–115. (Cited on page 125.)

Eastlake, D., Reagle, J., Imamura, T., Dillaway, B., and Simon, E. (2002a). XML Encryption Syntax and Processing. Technical report, W3C Recommendation. URL `http://www.w3.org/TR/xmlenc-core/`. (Cited on page 49.)

Eastlake, D., Reagle, J., Solo, D., Bartel, M., Boyer, J., Fox, B., LaMacchia, B., and Simon, E. (2002b). XML-Signature Syntax and Processing. Technical report, W3C Recommendation. URL `http://www.w3.org/TR/xmldsig-core/`. (Cited on pages 49 and 50.)

Eclipse (2007a). Eclipse - an open development platform. The Eclipse Foundation. URL `http://www.eclipse.org/`. (Cited on page 103.)

Eclipse (2007b). Eclipse Test & Performance Tools Platform Project. The Eclipse Foundation. URL http://www.eclipse.org/tptp/. (Cited on page 103.)

eDonkey (2006). eDonkey2000 Home Page. Meta Search Inc. URL http://www.edonkey2000.com/. (Cited on page 57.)

El-Sayed, A. and Black, J. (2006). Semantic-Based Context-Aware Service Discovery in Pervasive-Computing Environments. In *in Proceedings of IEEE Workshop on Service Integration in Pervasive Environments (SIPE), In conjunction with ICPS'06*. (Cited on page 139.)

Elbaz, L. (2002). Using Public Key Cryptography in Mobile Phones. Technical report, Discretix Technologies Ltd. URL http://www.discretix.com/PDF/Using%20Public%20Key%20Cryptography%20in%20Mobile%20Phones.pdf. (Cited on page 93.)

Elenius, D. (2003). *Service Discovery in Peer-to-Peer Networks*. Master's thesis, Linköpings universitet. URL http://www.ida.liu.se/%7Edaele/exjobb/files/thesis-final.pdf. (Cited on page 64.)

Elkarra, N. (2003). A Web Services Strategy for Mobile Phones. O'Reilly Inc. URL http://webservices.xml.com/pub/a/ws/2003/08/19/mobile.html. (Cited on page 22.)

Ellis, J. and Young, M. (2003). J2ME Web Services 1.0 - Final Draft (JSR 172). Technical Report 11, Sun Microsystems, Inc. (Cited on pages 18, 24 and 26.)

Endrei, M., Ang, J., Arsanjani, A., Chua, S., Comte, P., Krogdahl, P., Luo, M., and Newling, T. (2004). *Patterns: Service-Oriented Architecture and Web Services*. IBM Redbooks. URL http://www.redbooks.ibm.com/abstracts/sg246303.html. (Cited on page 7.)

van Engelen, R. A. and Gallivan, K. A. (2002). The gSOAP Toolkit for Web Services and Peer-To-Peer Computing Networks. In *in the proceedings of the 2nd IEEE International Symposium on Cluster Computing and the Grid (CCGrid2002)*. URL http://www.cs.fsu.edu/~engelen/ccgrid.pdf. (Cited on page 23.)

Ericsson (2003a). Enhanced Data Rates for GSM Evolution (EDGE) - Introduction of high-speed data in GSM/GPRS networks. Technical report, Ericsson AB. URL http://www.ericsson.com/technology/whitepapers/edge_wp_technical.pdf. (Cited on page 16.)

Ericsson, M. (2003b). A study of compression of XML-based messaging. Technical report, Växjö University. (Cited on pages 53 and 55.)

Ericsson, M. and Levenshteyn, R. (2003). On optimization of XML-based messaging. In *Second Nordic Conference on Web Services (NCWS 2003)*, pages 167–179. (Cited on pages 55 and 109.)

ETC (2007). New Media Review. European Travel Commission. URL http://www.etcnewmedia.com/review/default.asp?sectionid=10&overviewid=6. (Cited on pages 14 and 16.)

ETSI (1997). GSM Technical Specification, GSM 03.64: General Packet Radio Service (GPRS). Technical report, European Telecommunications Standards Institute (ETSI). (Cited on pages 16 and 31.)

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T. (1999). Hypertext Transfer Protocol – HTTP/1.1, RFC 2616. Technical report, Network Working Group. URL http://www.w3.org/Protocols/rfc2616/rfc2616.html. (Cited on pages 8 and 21.)

Forum Nokia (2004). Introduction To Web Services In Nokia Devices - Version 1.0. Technical report, Forum Nokia. URL http://sw.nokia.com/id/6fc34af5-877d-4877-bfef-c9144c023a7c/Nokia_Web_Services_v1_0_en.pdf. (Cited on page 18.)

Forum Nokia (2007a). Nokia Mobile Web Services Framework, Architecture, APIs, SDK. Forum Nokia. URL http://www.forum.nokia.com/main/resources/technologies/web_services/index.html. (Cited on page 27.)

Forum Nokia (2007b). Nokia Series 60 Platform. Forum Nokia. URL http://www.forum.nokia.com/main/platforms/s60/. (Cited on page 20.)

Forum Nokia (2007c). Nokia Series 80 Platform. Forum Nokia. URL http://www.forum.nokia.com/main/platforms/s80/. (Cited on page 20.)

Frank, S. P. (2004). Personal Java and Inferno for Today's Consumer Devices. *Java Developers Journal*, 3(3). URL http://www2.sys-con.com/ITSG/virtualcd/Java/archives/0303/frank/index.html. (Cited on page 20.)

Freier, A. O., Karlton, P., and Kocher, P. C. (1996). The SSL Protocol, Version 3.0. Technical report, Netscape Communications. URL http://wp.netscape.com/eng/ssl3/ssl-toc.html. (Cited on pages 49 and 78.)

loup Gailly, J. (2007). The GZip Home page. URL http://www.gzip.org/. (Cited on page 53.)

Gehlen, G. (2007). *Mobile Web Services - Concepts, Prototype, and Traffic Performance analysis*. Ph.D. thesis, RWTH Aachen University. (Cited on pages 2, 31 and 38.)

Gehlen, G., Aijaz, F., and Walke, B. (2006). Mobile Web Service Communication over UDP. In *Proceedings of the 64th IEEE Vehicular Technology Conference*. (Cited on page 31.)

Girardot, M. and Sundaresan, N. (2000). Millau: an encoding format for efficient representation and exchange of XML over the Web. In *Proceedings of the Nineth International World Wide Web Conference*. (Cited on page 55.)

Gokul, S. (2002). XML Encryption. Addison Wesley Professional. URL `http://www.awprofessional.com/articles/printerfriendly.asp?p=28801&rl=1`. (Cited on page 49.)

Google (2007). Google Maps. URL `http://maps.google.com/`. (Cited on page 16.)

Google Mobile (2007). Search, find and browse with Google on your mobile device. URL `http://www.google.com/mobile/search/index.html`. (Cited on page 16.)

Gospodnetic, O. and Hatcher, E. (2007). Meet Lucene. URL `http://www.developer.com/java/other/article.php/10936_3490471_1`. (Cited on page 66.)

Gottschalk, K., Graham, S., Kreger, H., and Snell, J. (2002). Introduction to Web Services architecture. *IBM Systems Journal: New Developments in Web Services and E-commerce*, 41(2):178–198. URL `http://researchweb.watson.ibm.com/journal/sj/412/gottschalk.html`. (Cited on pages xiii, 1 and 8.)

Greenberg, I. (2005). Memory breakthroughs will propel smartphone development - Transition from NOR to NAND flash memory a key. *Smartphone & Pocket PC magazine*. URL `http://www.pocketpcmag.com/_archives/Sep05/memory.aspx`. (Cited on page 15.)

Grimshaw, A. S., Wulf, W. A., and THE LEGION TEAM (1997). The Legion vision of a worldwide virtual computer. *Communications of the ACM*, 40(1):39–45. (Cited on page 57.)

Gudgin, M., Combs, H., Justice, J., Kakivaya, G., Lindsey, D., Orchard, D., Regnier, A., Schlimmer, J., Simpson, S., Tamura, H., Wright, D., and Wolf, K. (2004). SOAP-over-UDP. Technical report, BEA Systems Inc., Lexmark, Microsoft Corporation, Inc, and Ricoh. URL `http://specs.xmlsoap.org/ws/2004/09/soap-over-udp/soap-over-udp.pdf`. (Cited on page 30.)

Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J.-J., and Nielsen, H. F. (2003). SOAP Version 1.2 Part 2: Adjuncts. Technical report, World Wide Web Consortium Recommendation. URL `http://www.w3.org/TR/2003/REC-soap12-part2-20030624/`. (Cited on page 28.)

Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J.-J., Nielsen, H. F., Karmarkar, A., and Lafon, Y. (2007). SOAP Version 1.2 Part 1: Messaging Framework (Second Edition). Technical report, World Wide Web Consortium Recommendation. URL http://www.w3.org/TR/soap12-part1/. (Cited on page 9.)

Gudgin, M., Mendelsohn, N., Nottingham, M., and Ruellan, H. (2005). SOAP Message Transmission Optimization Mechanism. Technical report, W3C Recommendation. URL http://www.w3.org/TR/soap12-mtom/. (Cited on page 54.)

Gulbrandsen, A., Vixie, P., and Esibov, L. (2000). A DNS RR for specifying the location of services (DNS SRV) (RFC 2782). Technical report, Network Working Group. URL http://rfc.sunsite.dk/rfc/rfc2782.html. (Cited on page 123.)

Hajamohideen, S. H. (2003). *A Model for Web Service Discovery and Invocation in JXTA*. Master's thesis, Technical University Hamburg-Harburg. URL http://www.ti5.tu-harburg.de/publication/2003/Thesis/haja03/haja03.pdf. (Cited on pages xiii, 64 and 65.)

Halepovic, E. and Deters, R. (2003). The Costs of Using JXTA. In *The Third IEEE International Conference on Peer-to-Peer Computing*. (Cited on page 135.)

van Halteren, A. and Pawar, P. (2006). Mobile Service Platform: A Middleware for Nomadic Mobile Service Provisioning. In *2nd IEEE International Conference On Wireless and Mobile Computing, Networking and Communications (WiMob 2006)*. (Cited on pages xiii, 35 and 47.)

Hanson, J. J. (2005). ServiceMix as an enterprise service bus: Use ServiceMix 2.0 as a service-oriented message routing framework. JavaWorld.com. URL http://www.javaworld.com/javaworld/jw-12-2005/jw-1212-esb.html. (Cited on page 73.)

Harjula, E., Ylianttila, M., Ala-Kurikka, J., Riekki, J., and Sauvola, J. (2004). Plug-and-play application platform: Towards mobile peer-to-peer. In *Third International Conference on Mobile and Ubiquitous Multimedia (MUM2004)*, pages 63–69. (Cited on page 58.)

Harrison, R. (2003). *Symbian OS C++ for Mobile Phones*. John Wiley and Sons Ltd. (Cited on page 19.)

Helal, A., Haskell, B., Carter, J., Brice, R., Woelk, D., and Rusinkiewicz, M. (1999). *Any Time, Anywhere Computing; Mobile Computing Concepts and Technology*. Kluwer Academic Publishers. (Cited on page 14.)

Hemphill, D. (2004). Shout from the Hilltops with the J2ME Wireless Messaging API. devx.com. URL http://www.devx.com/wireless/Article/22266. (Cited on page 155.)

Heuer, J., Thienot, C., and Wollborn, M. (2002). *Introduction to MPEG-7: Multimedia Content Description Interface*, chapter Binary Format, pages 61–80. Jon Wiley and Son. (Cited on pages 53 and 55.)

Hodges, J. and Morgan, R. (2002). Lightweight Directory Access Protocol (v3): Technical Specification (RFC 3377). Technical report, Network Working Group. URL `http://www.rfc-editor.org/rfc/rfc3377.txt`. (Cited on page 123.)

Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., and Dean, M. (2004). SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Technical report, W3C Member Submission. URL `http://www.w3.org/Submission/SWRL/`. (Cited on page 97.)

Housley, R., Ford, W., Polk, W., and Solo, D. (1999). Internet X.509 Public Key Infrastructure Certificate and CRL Profile (RFC 2459). Technical report, Network Working Group. URL `http://www.ietf.org/rfc/rfc2459.txt`. (Cited on page 50.)

HP Labs (2007). HP Labs Semantic Web Research. URL `http://www.hpl.hp.com/semweb/`. (Cited on page 103.)

Hughes, J., Cantor, S., Hodges, J., Hirsch, F., Mishra, P., Philpott, R., and Maler, E. (2005). Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0. Technical report, OASIS Standard. URL `http://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf`. (Cited on page 52.)

IBM Corporation (2007a). IBM WebSphere Studio Device Developer. IBM developerWorks. URL `http://www-306.ibm.com/software/wireless/wsdd/`. (Cited on page 17.)

IBM Corporation (2007b). WebSphere Enterprise Service Bus. WebSphere Software. URL `http://www-306.ibm.com/software/integration/wsesb/`. (Cited on page 69.)

IBM Corporation (2008). IBM WebSphere MQ - Home page. WebSphere Software. (Cited on page 30.)

IBM Corporation and Microsoft Corporation (2002). Security in a Web Services world: A Proposed Architecture and Roadmap. IBM DeveloperWorks. URL `ftp://www6.software.ibm.com/software/developer/library/ws-secmap.pdf`. (Cited on pages xiii and 51.)

Ichikawa, K. (2002). The View of NTT DoCoMo on the Further development of Wireless Internet. In *Tokyo Mobile Round Table Conference*. (Cited on page 14.)

Imielinski, T., Viswanathan, S., and Badrinath, B. R. (1994). Power efficient filtering of data on air. In *Proceedings of the 4th international conference on extending database technology on Advances in database technology*, pages 245 – 258. Springer-Verlag. (Cited on page 18.)

IONA Technologies (2005). ESB: evolving beyond EAI. Technical report, IONA Technologies. URL http://www.itarchitects.ca/whitepaper/ESB-EvolvingBeyondEAI.pdf. (Cited on page 67.)

Ivanova, I. (2007). *Mobile Web Services for Collaborative Learning*. Master's thesis, RWTH Aachen University. (Cited on pages xv, 169, 170, 171, 172, 173, 176 and 179.)

java.net (2008). Project Open ESB. URL https://open-esb.dev.java.net/index.html. (Cited on page 70.)

Jayanti, V. B. K. and Hadley, M. (2005). SOAP with Attachments API for Java^{TM} (SAAJ) 1.3 - JSR 67. Technical report, Sun Microsystems, Inc. (Cited on page 150.)

JBoss (2007). JBoss Application Server. URL http://www.jboss.org/products/jbossas. (Cited on pages 73 and 94.)

Jena (2007). Jena, a semantic web framework for java. URL http://jena.sourceforge.net/. (Cited on pages 103 and 139.)

Jerz, M. (2006). Sony Ericsson P990i Review. URL http://my-symbian.com/uiq3/review_p990i.php. (Cited on page 109.)

Johansson, N. and Mollstedt, U. (2006). Revisiting Amit and Zott's model of value creation sources: The SymBelt Customer Center case. *Journal of Theoretical and Applied Electronic Commerce Research*, 1(3):16–27. (Cited on page 19.)

Johnson, D., Perkins, C., and Arkko, J. (2004). Mobility Support in IPv6 (RFC 3775). Technical report, Network Working Group. URL http://www.ietf.org/rfc/rfc3775.txt. (Cited on pages 32 and 121.)

Johnson, R. (2005). Introduction to the Spring Framework. TheServerSide.com. URL http://www.theserverside.com/tt/articles/article.tss?l=SpringFramework. (Cited on page 73.)

JXTA Community (2007a). JXTA-SOAP. java.net. URL https://soap.dev.java.net/. (Cited on page 64.)

JXTA Community (2007b). Project JXTA Community. java.net. URL https://jxta.dev.java.net/. (Cited on page 59.)

Kazaa (2007). Kazaa Home Page. URL http://www.kazaa.com/us/index.htm. (Cited on page 57.)

Keen, M., Bishop, S., Hopkins, A., Milinski, S., Nott, C., Robinson, R., Adams, J., Verschueren, P., and Acharya, A. (2004). *Patterns: Implementing an SOA using an Enterprise Service Bus*. IBM RedBooks. URL http://www.redbooks.ibm.com/redbooks/pdfs/sg246346.pdf. (Cited on pages 67 and 129.)

Kefali, U. (2004). *Development and performance evaluation of a simple object access protocol (SOAP) profile for block extensible exchange protocol (BEEP)*. Master's thesis, RWTH Aachen University. (Cited on page 30.)

Keon, D. M., Brewer, C., Carter, J., and Taggart, M. M. (2007). GSM and UMTS Security. URL http://ntrg.cs.tcd.ie/undergrad/4ba2.05/group7/index.html. (Cited on page 78.)

Kim, Y. K. and Prasad, R. (2006). *4G Roadmap and Emerging Communication Technologies*. Artech House Publishers. (Cited on page 32.)

Knudsen, J. (2002). Getting Started with JXTA for J2ME. Sun Developer Network. URL http://developers.sun.com/mobility/midp/articles/jxme/. (Cited on page 63.)

kSOAP (2007). kSOAP - An Open Source SOAP for the kVM. ObjectWeb. URL http://ksoap.objectweb.org/. (Cited on page 24.)

kSOAP2 (2007). kSOAP2 - An efficient, lean, Java SOAP library for constrained devices. SourceForge.net. URL http://sourceforge.net/projects/ksoap2. (Cited on page 25.)

Kurose, J. F. and Ross, K. W. (2001). *Computer Networking - A Top-Down Approach Featuring the Internet*. Pearson Education Addison Wesley. (Cited on pages xv and 190.)

Kwok, S. H., Lui, S., Cheung, R., Chan, S., and Yang, C. C. (2003). Searching Behavior in Peer-to-Peer Communities. In *International Conference on Information Technology: Computers and Communications (ITCC 03)*, page 130. IEEE Computer Society. (Cited on page 57.)

LA (2007). Liberty Alliance project. URL http://www.projectliberty.org/. (Cited on page 26.)

Lai, K. Y., Phan, T. K. A., and Tari, Z. (2005). Efficient SOAP Binding for Mobile Web Services. In *Proceedings of the IEEE Conference on Local Computer Networks 30th Anniversary (LCN'05)*. (Cited on page 31.)

Lai, X. (1992). On the design and security of block ciphers. *ETH Series in Information Processing, Hartung-Gorre Verlag Konstanz, Technische Hochschule Zurich*, 1. (Cited on page 85.)

Laukkanen, M. and Helin, H. (2003). Web Services in wireless networks: What happened to the performance. In *proceedings of the Int. Conf. on Web Services (ICWS '03)*, pages 278–284. CSREA Press. (Cited on page 108.)

Lee, C., Helal, A., Desai, N., Verma, V., and Arslan, B. (2003). Konark: A System and Protocols for Device Independent, Peer-to-Peer Discovery and Delivery of Mobile Services. *IEEE transactions on systems, man, and cybernetics - part a: systems and humans*, 33(6):682–696. (Cited on pages 124 and 125.)

Liefke, H. and Suciu, D. (1999). XMill: an efficient compressor for XML data. Technical report, University of Pennsylvania. (Cited on page 53.)

Liu, H., Pallikara, S., and Fox, G. (2005). Performance of Web Service Security. In *Proceedings of 13th Annual Mardi Gras Conference*. (Cited on page 79.)

LocatioNet (2007). amAze - Free GPS Navigation for your mobile phone! LocatioNet Solutions. URL `http://www.amazegps.com/`. (Cited on page 16.)

MacVittie, L. (2006). Review: ESB Suites - Make Way for the Enterprise Service Bus. Technical report, Network Computing. URL `http://www.networkcomputing.com/channels/appinfrastructure/showArticle.jhtml;jsessionid=CKJYQ2X3W1HY0QSNDBECKICCJUMEKJVN?articleID=181501276`. (Cited on page 70.)

Martin, B. and Jano, B. (1999). WAP Binary XML Content Format. Technical report, W3C NOTE. URL `http://www.w3.org/TR/wbxml/`. (Cited on page 54.)

Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., and Sycara, K. (2004). OWL-S: Semantic Markup for Web Services. Technical report, W3C Member Submission. URL `http://www.w3.org/Submission/OWL-S/`. (Cited on page 138.)

McGuinness, D. L. and van Harmelen, F. (2004). OWL Web Ontology Language - Overview. Technical report, W3C Recommendation. URL `http://www.w3.org/TR/owl-features/`. (Cited on pages 97 and 138.)

Meier, J., Mackman, A., Dunner, M., Vasireddy, S., Escamilla, R., and Murukan, A. (2003). Improving Web Application Security: Threats and Countermeasures. MSDN, Microsoft Corporation. URL `http://msdn2.microsoft.com/en-us/library/ms994921.aspx`. (Cited on page 78.)

Milojicic, D. S., Kalogeraki, V., Lukose, R., Nagaraja, K., Pruyne, J., Richard, B., Rollins, S., and Xu, Z. (2003). Peer-to-Peer Computing. Technical report, HP Laboratories Palo Alto. URL http://www.hpl.hp.com/techreports/2002/HPL-2002-57R1.pdf. (Cited on pages 4 and 57.)

Mitra, N. and Lafon, Y. (2007). SOAP Version 1.2 Part 0: Primer (Second Edition). Technical report, W3C Recommendation. (Cited on page 9.)

Mockapetris, P. V. and Dunlap, K. J. (1988). Development of the domain name system. In *Symposium proceedings on Communications architectures and protocols*, pages 123 – 133. ACM Press. (Cited on page 123.)

Moses, T. (2005). eXtensible Access Control Markup Language (XACML) Version 2.0. Technical report, OASIS Standard. URL http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf. (Cited on page 52.)

MSDN (2007a). .NET Compact Framework. Microsoft Corporation. URL http://msdn2.microsoft.com/en-us/netframework/aa497273.aspx. (Cited on page 22.)

MSDN (2007b). .NET Framework. Microsoft Corporation. URL http://msdn2.microsoft.com/en-us/netframework/default.aspx. (Cited on page 22.)

MSDN (2007c). Web Services Enhancements (WSE). Microsoft Corporation. URL http://msdn2.microsoft.com/en-us/webservices/aa740663.aspx. (Cited on page 29.)

Naumenko, A. (2007a). *Semantic-Based Access Control in Business Networks*. Ph.D. thesis, University of jyväskylä. (Cited on page 103.)

Naumenko, A. (2007b). Semantics-Based Access Control: Ontologies and Feasibility Study of Policy Enforcement Function. In *Third International Conference on Web Information Systems and Technologies (WEBIST 2007)*, volume Internet Technologies, pages 150–155. INSTICC Press. (Cited on page 97.)

Naumenko, A., Katasonov, A., and Terziyan, V. (2007a). A Security Framework for Smart Ubiquitous Industrial Resources. In J. Müller and K. Mertins, editors, *Proceedings of the 3rd International Conference on Interoperability for Enterprise Software and Applications (IESA-07)*, page 13. (Cited on page 163.)

Naumenko, A. and Luostarinen, K. (2006). Access Control Policies in (Semantic) Service-Oriented Architecture. In S. S. and S. Y., editors, *Proceedings of the SEMANTICS 2006*, pages 49–62. Austrian Computer Society. (Cited on page 97.)

Naumenko, A., Srirama, S., Terziyan, V., and Jarke, M. (2007b). Semantics-Based Access Control for Mobile Web Services. *International Journal on Semantic Web and Information systems, Special Issue on Mobile services and Ontologies (Submitted for Publication)*. (Cited on pages xiv, 98, 100, 103, 105 and 123.)

NAVSTAR (1995). Global Positioning System Standard Positioning Service Signal Specification - 2nd Edition. Technical report, U.S. Coast Guard Navigation Center. URL `http://www.navcen.uscg.gov/pubs/gps/sigspec/gpssps1.pdf`. (Cited on page 40.)

NAVSTAR (2007). USNO NAVSTAR Global Positioning System. NAVSTAR GPS Operations. URL `http://tycho.usno.navy.mil/gpsinfo.html`. (Cited on page 40.)

Neuman, B. C. and Ts'o, T. (1994). Kerberos: An Authentication Service for Computer Networks. *IEEE Communications Magazine*, 32(9):33–38. (Cited on page 50.)

NIST (1995). Secure Hash Standard - FIPS PUB 180-1. Technical report, National Institute of Standards and Technology. URL `http://www.itl.nist.gov/fipspubs/fip180-1.htm`. (Cited on pages 50 and 93.)

NIST (1999). Data Encryption Standard (DES). Technical report, National Institute of Standards and Technology. URL `http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf`. (Cited on pages 49 and 85.)

Novak, L. and Svensson, M. (2001). MMS - building on the success of SMS. *Ericsson Review No. 3*, pages 102–109. URL `http://www.ericsson.com/ericsson/corpinfo/publications/review/2001_03/files/2001031.pdf`. (Cited on page 39.)

NTT DoCoMo (2003). NTT DoCoMo Unveils 505i Series i-mode compatible Mobile Phones Equipped for Macromedia Flash and Enhanced Java-Based Applications. NTT DoCoMo Press Releases. URL `http://www.nttdocomo.com/pr/2003/000946.html`. (Cited on page 15.)

NTT DoCoMo (2007a). Data Communications with FOMA and PC. URL `http://www.nttdocomo.co.jp/english/service/data/`. (Cited on page 32.)

NTT DoCoMo (2007b). i-mode Business Model. URL `http://www.nttdocomo.com/services/imode/business/index.html`. (Cited on page 17.)

NTT DoCoMo (2007c). i-mode Technology. URL `http://www.nttdocomo.com/technologies/present/imodetechnology/index.html`. (Cited on page 16.)

NTT DoCoMo (2007d). NTT DoCoMo: HOME. URL `http://www.nttdocomo.com/`. (Cited on page 16.)

OMA (2004). Open mobile alliance overview. Technical report, Open Mobile Alliance Group. URL http://www.openmobilealliance.org/docs/OMAShortPaper_May2004v.1.doc. (Cited on pages 17 and 27.)

OMA (2006a). Mobile Web Services Requirements - Version 1.1. Technical report, Open Mobile Alliance Group. URL http://www.openmobilealliance.org/release_program/docs/OWSER/V1_1-20060328-A/OMA-RD-OWSER-V1_1-20060328-A.pdf. (Cited on pages 17 and 27.)

OMA (2006b). OMA Web Services Enabler (OWSER): Core Specifications - Version 1.1. Technical report, Open Mobile Alliance Group. URL http://www.openmobilealliance.org/release_program/docs/OWSER/V1_1-20060328-A/OMA-TS-OWSER_Core_Specification-V1_1-20060328-A.pdf. (Cited on page 27.)

OMG (2004). Common Object Request Broker Architecture (CORBA): Core Specification. Technical report, Object Management Group. URL http://www.omg.org/docs/formal/04-03-12.pdf. (Cited on pages 7 and 52.)

Open Source Initiative (2008). Open Source Initiative (OSI) - Home page. (Cited on page 186.)

O'Reilly, T. (2005). What Is Web 2.0 - Design Patterns and Business Models for the Next Generation of Software. URL http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html. (Cited on page 163.)

Ozzie, J. and Burton, A. (2003). Delivering Collaboration Services to Information Workers: Extending SharePoint with Groove. In *Power Point presentation at Microsoft SharePoint products and technologies Developer's conference*. (Cited on page 57.)

Papanikolaou, K. and Mavromoustakos, S. (2006). Critical success factors for the development of mobile learning applications. In *IMSA'06: Proceedings of the 24th IASTED international conference on Internet and multimedia systems and applications*, pages 19–24. ACTA Press, Anaheim, CA, USA. ISBN 0-88986-564-7. (Cited on page 165.)

Pashtan, A. (2005). *Mobile Web Services*. Cambridge University Press. (Cited on pages 36 and 162.)

Pawar, P., Srirama, S., van Beijnum, B.-J., and van Halteren, A. (2007). A Comparative Study of Nomadic Mobile Service Provisioning Approaches. In *International Conference And Exhibition On Next Generation Mobile Applications, Services And Technologies (NGMAST 2007)*, pages 277–286. IEEE Computer Society. (Cited on pages xiii, 47 and 48.)

Pendyala, K. K. (2006). *Security Aspects Analysis in Mobile Web Service Provisioning*. Master's thesis, RWTH Aachen University. (Cited on pages xiv, 88 and 95.)

Perera, A. (2007). WSO2 ESB Performance Testing Round 2. URL `http://wso2.org/library/2259`. (Cited on page 157.)

PolarLake (2005). Understanding the ESB: What it is, why it matters, and how to choose one (White Paper). Technical report, PolarLake. URL `http://www.polarlake.com/files/esb.pdf`. (Cited on page 68.)

Pouwelse, J., Garbacki, P., Epema, D., and Sips, H. (2005). The bittorrent P2P file-sharing system: Measurements and analysis. In *Proceedings of 4th International Workshop on Peer-to-Peer Systems (IPTPS'05)*. (Cited on page 57.)

Protégé (2007). Welcome to Protégé - Home page. URL `http://protege.stanford.edu/`. (Cited on page 103.)

Prud'hommeaux, E. and Seaborne, A. (2007). SPARQL Query Language for RDF. Technical report, W3C Candidate Recommendation. URL `http://www.w3.org/TR/rdf-sparql-query/`. (Cited on page 103.)

Pulkkinen, M., Naumenko, A., and Luostarinen, K. (2007). Managing Information Security in a Business Network of Machinery Maintenance Services Business - Enterprise Architecture as a Coordination Tool. *Special Issue on Methodology of Security Engineering for Industrial Security Management Systems, Journal of Systems and Software, ELSEVIER (In Press)*. (Cited on page 19.)

Qu, C. and Nejdl, W. (2004). Interacting the Edutella/JXTA Peer-to-Peer Network with Web Services. In *Proceedings of the 2004 International Symposium on Applications and the Internet (SAINT'04)*. (Cited on page 64.)

QuickLogic (2007). Using Hardware Acceleration to Optimize Software-Based Embedded Systems, White Paper (Revision B). Technical report, QuickLogic Corporation. URL `http://www.quicklogic.com/images/Optimizing_Embedded_Systems_WP.pdf`. (Cited on page 92.)

Raghunathan, V., Schurgers, C., Park, S., and Srivastava, M. B. (2002). Energy-aware wireless microsensor networks. *IEEE Signal Processing Magazine*, 19(2):40–50. (Cited on page 116.)

Redl, S. M., Weber, M. K., and Oliphant, M. W. (1995). *An Introduction to GSM*. Artech House Publishers. (Cited on page 31.)

Rescorla, E. (2000). HTTP Over TLS, RFC 2818. Technical report, IETF. URL `http://www.ietf.org/rfc/rfc2818.txt`. (Cited on page 78.)

Ripeanu, M., Foster, I., and Iamnitchi, A. (2002). Mapping the Gnutella Network: Properties of Large-Scale Peer-to-Peer Systems and Implications for System Design. *IEEE Internet Computing Journal*, 6(1):51Ű57. (Cited on page 57.)

Rivest, R. (1992). RFC 1321 - The MD5 Message-Digest Algorithm. Technical report, Network Working Group. URL http://www.faqs.org/rfcs/rfc1321.html. (Cited on pages 50 and 93.)

Rivest, R., Shamir, A., and Adleman, L. (1978). A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21(2):120–126. (Cited on page 83.)

Rodriguez, S. (2002). XML optimization. URL http://www.codeproject.com/soap/betterxml.asp. (Cited on page 54.)

Rollman, R. and Schneider, J. (2004). Mobile web services. In *XML 2004 Proceedings by SchemaSoft*. URL http://www.idealliance.org/proceedings/xml04/papers/73/MobileWebServices.pdf. (Cited on page 14.)

RSA Labs (2007a). Advanced Encryption Standard (AES). URL http://www.rsasecurity.com/rsalabs/node.asp?id=2234. (Cited on page 83.)

RSA Labs (2007b). Cryptographic technologies. URL http://www.rsa.com/rsalabs/node.asp?id=2212. (Cited on pages 49 and 83.)

RSA Labs (2007c). Digital Signature Standard (DSS). URL http://www.rsasecurity.com/rsalabs/node.asp?id=2239. (Cited on page 83.)

RSA Labs (2007d). Triple Digital Encryption Standard. URL http://www.rsasecurity.com/rsalabs/node.asp?id=2231. (Cited on page 83.)

Rysavy, P. (1998). General Packet Radio Service (GPRS). *GSM Data Today online journal*. URL http://www.rysavy.com/Articles/GPRS/GPRS.htm. (Cited on pages 16 and 31.)

Sandoz, P., Pericas-Geertsen, S., Kawaguchi, K., Hadley, M., and Pelegri-Llopart, E. (2003). Fast Web Services. URL http://java.sun.com/developer/technicalArticles/WebServices/fastWS/. (Cited on page 53.)

Sandoz, P., Triglia, A., and Pericas-Geertsen, S. (2004). Fast Infoset. URL http://java.sun.com/developer/technicalArticles/xml/fastinfoset/. (Cited on page 53.)

Schneider, J. (2001). Convergence of Peer and Web Services. OpenP2P.com. URL http://www.openp2p.com/lpt/a/1047. (Cited on page 58.)

Schulte, R. W. (2007). The Enterprise Service Bus: Communication Backbone for SOA. Gartner Inc. URL http://www.gartner.com/DisplayDocument?id=504645. (Cited on page 67.)

Seward, J. (2005). bzip2 and libbzip2, version 1.0.3 - A program and library for data compression. URL http://www.bzip.org/1.0.3/bzip2-manual-1.0.3.html. (Cited on page 54.)

Shirky, C., Truelov, K., Dornfest, R., and Gonze, L. (2001). *P2P Networking Overview*. O'Reilly. (Cited on pages 39 and 56.)

da Silva, R. (2007). eSOAP - Features. URL http://esoap.ultimodule.com/bin/esoap/templates/splash.asp?NC=6825X. (Cited on page 23.)

Silva, R. D. (2001). SOAP and Embedded Systems - Draft 0.4. Technical report, ConnectTel, Inc. (Cited on page 23.)

Skype (2007). Skype Home page. URL http://www.skype.com/helloagain.html. (Cited on page 57.)

SourceID (2004). *SourceID Liberty 2.0 Beta - User's Guide*. Ping Identity Corporation. URL http://www.sourceid.org/docs/upload/ID-FF-1-2-Java-Toolkit-2-0-Beta-User-s-Guide.pdf. (Cited on page 94.)

Srirama, S. (2004). *Concept, implementation and performance testing of a mobile Web Service provider for smart phones*. Master's thesis, RWTH Aachen, Germany. (Cited on pages 2, 35, 36 and 45.)

Srirama, S. (2006). Publishing and Discovery of Mobile Web Services in Peer to Peer Networks. In *Proceedings of First International Workshop on Mobile Services and Personalized Environments (MSPE'06)*, volume P-102, pages 15–28. Lecture Notes in Informatics, GI. (Cited on pages xiv, 119, 120, 121 and 127.)

Srirama, S. (2007). Mobile Web Service provisioning. URL http://www-i5.informatik.rwth-aachen.de/lehrstuhl/staff/srirama/MWSP.html. (Cited on page 35.)

Srirama, S., Jarke, M., and Prinz, W. (2006a). Mobile Host: A feasibility analysis of mobile Web Service provisioning. In *4th International Workshop on Ubiquitous Mobile Information and Collaboration Systems, UMICS 2006, a CAiSE'06 workshop*, pages 942–953. (Cited on pages xiii, xv, 41, 43, 44, 45, 122 and 163.)

Srirama, S., Jarke, M., and Prinz, W. (2007a). Security analysis of mobile web service provisioning. *International Journal of Internet Technology and Secured Transactions (IJITST)*, 1(1/2):151–171. (Cited on pages xiii, xiv, 37, 78, 79, 80, 82, 86, 87, 89 and 90.)

Srirama, S., Jarke, M., Prinz, W., and Pendyala, K. (2006b). Security aware mobile Web Service provisioning. In *Proceedings of International Conference for Internet Technology and Secured Transactions (ICITST-2006)*, pages 48–56. e.Centre for Infonomics. (Cited on page 79.)

Srirama, S. N., Jarke, M., and Prinz, W. (2006c). A mediation framework for mobile Web Service provisioning. In *2006 Middleware for Web Services (MWS 2006) Workshop @ 10th IEEE International Enterprise Distributed Object Computing Conference*, page 14. IEEE Computer Society. ISBN 0-7695-2743-4. doi:http://doi.ieeecomputersociety.org/10.1109/EDOCW.2006.9. (Cited on pages xv, 129 and 147.)

Srirama, S. N., Jarke, M., and Prinz, W. (2006d). Mobile Web Service provisioning. In *AICT-ICIW '06: Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services*, page 120. IEEE Computer Society. ISBN 0-7695-2522-9. (Cited on pages xiii, 2, 36 and 38.)

Srirama, S. N., Jarke, M., and Prinz, W. (2007b). Mobile Web Service Discovery in Peer to Peer Networks. In *5th International Workshop on Ubiquitous Mobile Information and Collaboration Systems, UMICS 2007, a CAiSE'07 workshop*, pages 531–542. (Cited on pages xiv and 135.)

Srirama, S. N., Jarke, M., and Prinz, W. (2007c). Mobile Web Services Mediation Framework. In *Middleware for Service Oriented Computing (MW4SOC) Workshop @ 8th International Middleware Conference 2007 (In Print)*. ACM Press. (Cited on pages xv, 148 and 158.)

Srirama, S. N., Jarke, M., and Prinz, W. (2007d). A performance evaluation of mobile web services security. In *Third International Conference on Web Information Systems and Technologies (WEBIST 2007)*, pages 386–392. INSTICC Press. (Cited on page 90.)

Srirama, S. N., Jarke, M., and Prinz, W. (2008a). MWSMF: A Mediation Framework Realizing Scalable Mobile Web Service Provisioning. In *International Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications (Mobilware 2008) (In Print)*. (Cited on pages xiv, xv, 115, 151, 153 and 159.)

Srirama, S. N., Jarke, M., Prinz, W., and Zhu, H. (2008b). Scalable Mobile Web Service Discovery in Peer to Peer Networks. In *IEEE Third International Conference on Internet and Web Applications and Services (ICIW 2008) (Accepted for publication)*. (Cited on pages xiv, xv, 123, 134, 136 and 137.)

Srirama, S. N. and Naumenko, A. (2007). Secure Communication and Access Control for Mobile Web Service Provisioning. In *Proceedings of the First International Conference on Security of Information and Networks (SIN 2007)*. Trafford Publishing, Canada. (Cited on pages xiii, xiv, 18, 81, 96 and 102.)

Sun Microsystems (2000). J2ME Building Blocks for Mobile Devices - White Paper on KVM and the Connected, Limited Device Configuration (CLDC). Technical report, Sun Microsystems, Inc. URL http://java.sun.com/products/cldc/wp/KVMwp.pdf. (Cited on page 21.)

Sun Microsystems (2001). Jini Architecture Specification - Version 1.2. Technical report, Sun Microsystems, Inc. URL http://www.sun.com/software/jini/specs/jini1.2html/jini-title.html. (Cited on pages 46 and 48.)

Sun Microsystems (2005). CDC: Java^{TM} platform technology for connected devices. Technical report, Sun Microsystems, Inc. URL http://java.sun.com/products/cdc/wp/cdc-whitepaper.pdf. (Cited on page 21.)

Sun Microsystems (2007a). J2ME Web Services APIs (WSA), JSR 172. Sun Developer Network. URL http://java.sun.com/products/wsa/. (Cited on page 24.)

Sun Microsystems (2007b). Java 2 standard edition development kit 1.5. Sun Developer Network. URL http://java.sun.com/j2se/1.5.0/. (Cited on page 103.)

Sun Microsystems (2007c). The Java ME Device Table. Sun Developer Network. URL http://developers.sun.com/mobility/device/pub/device/list.do?sort=manufacturer&page=33. (Cited on pages 21 and 27.)

Sun Microsystems (2007d). Java^{TM} 2 Platform, Micro Edition (J2ME^{TM}) Web Services Specification - Datasheet. Technical report, Sun Microsystems, Inc. URL http://java.sun.com/j2me/docs/j2me_jsr172.pdf. (Cited on page 26.)

Sun Microsystems (2007e). PersonalJava. Sun Developer Network. URL http://java.sun.com/products/personaljava/. (Cited on page 20.)

Sun Microsystems (2007f). Sun Java Wireless Toolkit for CLDC. Sun Developer Network. URL http://java.sun.com/products/sjwtoolkit/. (Cited on page 17.)

Sun Microsystems (2007g). The Java ME Platform - the Most Ubiquitous Application Platform for Mobile Devices. Sun Developer Network. URL http://java.sun.com/javame/index.jsp. (Cited on page 21.)

Sun Microsystems (2007h). The Jini^{TM} Technology Surrogate Architecture Overview. Technical report, Sun Microsystems, Inc. URL https://surrogate.dev.java.net/doc/overview.pdf. (Cited on page 47.)

Sun Microsystems (2007i). Wireless Messaging API (WMA); JSR 120, JSR 205. Sun Developer Network. URL http://java.sun.com/products/wma/index.jsp. (Cited on page 155.)

Sun Microsystems (2008). Java Development Kit (JDK), v 1.1 - Archive Home page. Sun Developer Network. (Cited on page 20.)

Symbian (2007). Symbian OS - The mobile operating system. URL `http://www.symbian.com/`. (Cited on page 20.)

Tang, K., Chen, S., Levy, D., Zic, J., and Yan, B. (2006). A Performance Evaluation of Web Services Security. In *10th IEEE International Enterprise Distributed Object Computing Conference (EDOC'06)*, pages 67–74. (Cited on page 79.)

Tate, B. (2005). Secrets of lightweight development success, Part 1: Core principles and philosophies. IBM DeveloperWorks. URL `http://www.ibm.com/developerworks/library/os-lightweight1/`. (Cited on page 73.)

Tate, B. A. and Gehtland, J. (2005). *Spring: A Developer's Notebook*. O'Reilly. (Cited on page 149.)

Ten-Hove, R. (2006). JBI Components: Part 1 (Theory). URL `https://open-esb.dev.java.net/public/pdf/JBI-Components-Theory.pdf`. (Cited on page 72.)

Ten-Hove, R. and Walker, P. (2005). Java^{TM} Business Integration (JBI) 1.0 -JSR 208 Final Release. Technical report, Sun Microsystems, Inc. (Cited on page 70.)

Thomas, N. and Buckley, W. (2003). The Enterprise Service Bus: A Developer-Friendly Integration Engine. *WebServices Journal*, 3(10):30. (Cited on page 68.)

TomTom (2007). TomTom, portable GPS car navigation systems. URL `www.tomtom.com/`. (Cited on page 16.)

Tourzan, J. and Koga, Y. (2006). Liberty ID-WSF Web Services Framework Overview - Version: 2.0. Technical report, Liberty Alliance Project. URL `http://www.projectliberty.org/liberty/content/download/889/6243/file/liberty-idwsf-overview-v2.0.pdf`. (Cited on pages 17 and 26.)

Traversat, B., Abdelaziz, M., Doolin, D., Duigou, M., Hugly, J.-C., and Pouyoul, E. (2003). Project JXTA-C: Enabling a Web of Things. In *36th Annual Hawaii International Conference on System Sciences (HICSS'03)*, page 282. IEEE Computer Society. (Cited on page 63.)

UIQ (2007). UIQ Technology Home. URL `http://www.uiq.com/`. (Cited on page 20.)

Umtsworld (2002). Overview of the Universal Mobile Telecommunication System. UMTS world. URL `http://www.umtsworld.com/technology/overview.htm`. (Cited on pages 16 and 32.)

University College Cork, Ireland (2007). SUMI- the de facto industry standard evaluation questionnaire for assessing quality of use of software by end users. URL http://sumi.ucc.ie/index.html. (Cited on page 181.)

UPnP Forum (2003). Universal Plug and Play Device Architecture. Technical report, UPnP Forum. URL http://www.upnp.org/resources/documents/CleanUPnPDA101-20031202s.pdf. (Cited on page 125.)

U.S. Census Bureau (2007). North American Industry Classification System (NAICS). URL http://www.census.gov/epcd/www/naics.html. (Cited on page 133.)

Verma, M. (2003). XML Security: Implement security layers, Part 2. IBM DeveloperWorks. URL http://www-128.ibm.com/developerworks/xml/library/x-seclay2/. (Cited on page 50.)

Vinoski, S. (2005). Java Business Integration. *IEEE Internet Computing*, pages 89–91. URL http://ieeexplore.ieee.org/iel5/4236/31457/01463167.pdf?arnumber=1463167. (Cited on page 72.)

Waterhouse, S., Doolin, D. M., Kan, G., and Faybishenko, Y. (2002). Distributed Search in P2P Networks. *IEEE Internet Computing*, pages 68–73. (Cited on page 128.)

Wenger, E. (1998). *Communities of practice: learning, meaning, and identity*. Cambridge University Press. (Cited on page 161.)

Werden, S., Evans, C., and Goodner, M. (2003). WS-I Usage Scenarios. Technical report, Web Services Interoperability Organization. URL http://www.ws-i.org/SampleApplications/SupplyChainManagement/2003-12/UsageScenarios-1.01.pdf. (Cited on page 14.)

Whiteside, A. (2007). OGC Web Services Common Specification. Technical report, Open Geospatial Consortium Inc. (Cited on page 19.)

Wilson, B. J. (2002). *JXTA*. New Riders Publishing. (Cited on pages xiii, 4 and 60.)

Wingfoot (2007a). Wingfoot SOAP 1.06 - User Guide. Technical report, Wingfoot Software. URL http://www.wingfoot.com/docs/WSOAP_User_Guide1_06.html. (Cited on page 24.)

Wingfoot (2007b). Wingfoot SOAP Client. URL http://www.wingfoot.com/products.html. (Cited on page 24.)

WS-I (2004). Interoperability: Ensuring the Success of Web Services - An Overview to WS-I. Web Services Interoperability organization (WS-I). URL http://www.ws-i.org/docs/20041130.introduction.ppt. (Cited on page 13.)

WS-Talk (2007). WS-Talk Project. URL `http://thames.cs.rhul.ac.uk/~wstalk/`. (Cited on pages xiii, 65 and 66.)

WSO2 (2007). WSO2 ESB Benchmark. URL `https://www-lk.wso2.com/~asankha/benchmark/`. (Cited on page 157.)

Yang, X., Bouguettaya, A., Medjahed, B., Long, H., and He, W. (2003). Organizing and Accessing Web Services on Air. *IEEE transactions on systems, man, and cybernetics - part a: systems and humans*, 33(6):742–757. (Cited on pages 18 and 125.)

Zhu, H. (2008). *Scalability of P2P Based Mobile Web Services Discovery*. Master's thesis, RWTH Aachen University. (Cited on page 138.)

Ziv, J. and Lempel, A. (1977). A Universal Algorithm for Sequential Data Compression. *IEEE Transactions On Information Theory*, 23(3):337–344. (Cited on page 53.)

Index

Scientific Publications

Journal Publications

Srirama, S., Jarke, M., and Prinz, W. (2007). Security Analysis of Mobile Web Service Provisioning. *International Journal of Internet Technology and Secured Transactions (IJITST)*, 1(1/2):151-171. Inderscience Publishers.

Naumenko, A., Srirama, S., Terziyan, V., and Jarke, M. (2007). Semantics-Based Access Control for Mobile Web Services. *International Journal on Semantic Web and Information systems, Special Issue on Mobile services and Ontologies. (Submitted for Publication).*

Chatti, M. A., Srirama, S., Ivanova, I., Jarke, M. (2008). The MobileHost CoLearn System: Mobile Social Software for Collaborative Learning. *International Journal of Mobile Learning and Organisation (IJMLO), Special Issue on: "Developing Themes in Mobile Learning". (Accepted for publication).*

Conference Publications

Srirama, S. N., Jarke, M., Prinz, W., and Zhu, H. (2008). Scalable Mobile Web Service Discovery in Peer to Peer Networks. *IEEE Third International Conference on Internet and Web Applications and Services (ICIW 2008)*. IEEE Computer Society.

Srirama, S. N., Jarke, M., and Prinz, W. (2008). MWSMF: A Mediation Framework Realizing Scalable Mobile Web Service Provisioning. *International Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications (Mobilware 2008)*. ACM Press.

Pawar, P., Srirama, S., van Beijnum, B. J., and van Halteren, A. (2007). A Comparative Study of Nomadic Mobile Service Provisioning Approaches. *International Conference And Exhibition On Next Generation Mobile Applications, Services And Technologies (NGMAST 2007)*, pages 277-286. IEEE Computer Society.

Srirama, S. N., and Naumenko, A. (2007). Secure Communication and Access Control for Mobile Web Service Provisioning. *Proceedings of the First International Conference on Security of Information and Networks (SIN 2007)*. Trafford Publishing, Canada.

Srirama, S. N., Jarke, M., and Prinz, W. (2007). A Performance Evaluation of Mobile Web Services Security. *Third International Conference on Web Information Systems and Technologies (WEBIST 2007)*, pages 386-392. INSTICC Press.

Srirama, S., Jarke, M., Prinz, W., and Pendyala, K. (2006). Security Aware Mobile Web Service Provisioning. *Proceedings of International Conference for Internet Technology and Secured Transactions (ICITST-2006)*, pages 48-56. e.Centre for Infonomics.

Srirama, S. N., Jarke, M., and Prinz, W. (2006). Mobile Web Service provisioning. *AICT-ICIW '06: Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services*, page 120. IEEE Computer Society.

Other Publications

Srirama, S. N., Jarke, M., and Prinz, W. (2007). Mobile Web Services Mediation Framework. *In Middleware for Service Oriented Computing (MW4SOC) Workshop @ 8th International Middleware Conference 2007 (In Print)*. ACM Press.

Cao, Y., Klamma, R., Srirama, S., and Wang, K. (2007). The Mobile Interfaces for Geo-Hypermedia Databases. *In Proc. of Workshop on Mobile and Networking Technologies for Social Applications (MONET 2007) @ OTM 2007 Federated Conferences (In Print)*. LNCS, Springer.

Srirama, S. N., Jarke, M., and Prinz, W. (2007). Mobile Web Service Discovery in Peer to Peer Networks. *In 5th International Workshop on Ubiquitous Mobile Information and Collaboration Systems, UMICS 2007, a CAiSE'07 workshop*, pages 531-542.

Srirama, S. (2006). Publishing and Discovery of Mobile Web Services in Peer to Peer Networks. *In Proceedings of First International Workshop on Mobile Services and Personalized Environments (MSPE'06)*, volume P-102, pages 15-28. Lecture Notes in Informatics, GI.

Chatti, M. A., Srirama, S., Kensche, D., and Cao, Y. (2006). Mobile Web Services for Collaborative Learning. *In Fourth IEEE International Workshop on Wireless, Mobile and Ubiquitous Technology in Education - (WMUTE'06)*, pages 129-133.

Cao, Y., Srirama, S., Chatti, M. A., and Klamma, R. (2006). Mobile Social Software for Cultural Heritage Management. *In Proc. of Workshop on Mobile and Networking Technologies for Social Applications (MONET 2006) @ OTM 2006 Federated Conferences*, pages 955-964. LNCS 4277, Springer.

Srirama, S., Jarke, M., and Prinz, W. (2006). A Mediation Framework for Mobile Web Service Provisioning. *Proceedings of 2006 Middleware for Web Services (MWS 2006) Workshop @ 10th International IEEE EDOC Conference "The Enterprise Computing Conference"*, pages 942-953. IEEE Computer Society.

Srirama, S., Jarke, M., and Prinz, W. (2006). Mobile Host: A Feasibility Analysis of Mobile Web Service Provisioning. *In 4th International Workshop on Ubiquitous Mobile Information and Collaboration Systems, UMICS 2006, a CAiSE'06 workshop*, pages 942-953.

Srirama, S., Kakumani, R., Aggarwal, A., and Pawar, P. (2006). Effective Testing Principles for the Mobile Data Services Applications. *IEEE International Workshop on Software for Wireless Communications and Applications (SOFTWIM 2006) @ The First International Conference on Communication Systems Software and Middleware (COMSWARE 2006)*, pages 1-5. IEEE.

Curriculum Vitae

Name:	Satish Narayana Srirama
Date of birth:	23rd August 1978
Place of birth:	Peddapadmapuram, A.P., India
Nationality:	Indian

03/1992 - 03/1993	Studied S.S.C. (Secondary School Certificate) at V.T.College, Visakhapatnam, India
04/1993 - 05/1995	Studied Intermediate at Raghu Junior College, Visakhapatnam, India
06/1995 - 04/1999	Studied **B.Tech** (Bachelor of Technology) in *Computer Science and Systems Engineering* at Andhra University, Visakhapatnam, India
07/1999 - 06/2000	Worked as *Software Programmer* at Waltair Info Solutions (P) Ltd, Visakhapatnam, India
06/2000 - 12/2001	Worked as *Software Engineer* at RapidSoft Networks Ltd, Hyderabad, India
01/2002 - 09/2002	Worked as *Software Engineer* at B2B Software Technologies Ltd, Hyderabad, India
10/2002 - 07/2004	Studied **M.Sc.** (Master of Sciences) in *Software Systems Engineering* at RWTH Aachen University, Germany
11/2003 - 03/2008	Worked as *Research Helper* at Ericsson GmbH, Eurolab R&D, Aachen, Germany
08/2004 - 10/2004	Worked as *Software Engineer* at Nevis Networks (India) Private Ltd, Pune, India
Since 08/2004	Working as *Research Assistant* at RWTH Aachen University, Germany