A Comparative Study of Nomadic Mobile Service Provisioning Approaches

Pravin Pawar[1], Satish Srirama[2], Bert-Jan van Beijnum[1], Aart van Halteren[3]
[1] *Architecture and Services for Networked Applications Group*
Department of Computer Science
University of Twente, The Netherlands.
{p.pawar, b.j.f.vanbeijnum}@utwente.nl
[2] *Information Systems Group*
RWTH Aachen University, Germany.
srirama@cs.rwth-aachen.de
[3] *Philips Research, Eindhoven, The Netherlands.*
aart.van.halteren@philips.com

Abstract

In today's world of pervasive computing, the mobile devices are enriched with a variety of features and being used as a personal information delivery platform. The increased processing, storage and communication capabilities of these devices combined with the latest developments in the area of Service Oriented Architecture enables a new class of services, called Nomadic Mobile Services (NMS). Recent research has resulted in different NMS provisioning approaches; each one employs/defines a different architecture and addresses a different mix of issues. This paper provides a comparative study of three NMS provisioning approaches based on their architectural design, development choices and prototyped applications. Each approach has its own merits considering the applications they aim at. However, in the future, a solution which uses web services for better interoperability and employes proxy approach for better QoS could be a possible technical design.

1. Introduction

Over the past two decades, developments in the area of distributed computing such as *Remote Procedure Calls, Object Middleware* and *Component-based Middleware* have lead to *Service Oriented Architecture* (SOA) [1]. Nowadays, SOA is a popular choice for the application development because it facilitates the development, deployment and usage of (composite) services which are well defined, loosely coupled, flexible, reusable and have implementation independent interfaces.

Today, advanced mobile devices such as smart phones and PDAs are characterized by higher processing power, increased memory and availability of multiple network interfaces such as Wi-Fi, GSM, UMTS which enable them to connect to the Internet. Mobility, portability and connectivity have enabled convenient personal applications, turning mobile devices into a *personal information delivery platform*. Until recently, the research in the SOA community has been focused on the use of mobile devices in the role of a service consumer, e.g. [2, 3]. However, a mobile device in the role of a *service provider* enables, amongst others, entirely new scenarios and end-user services. This *paradigm shift* from the role of service consumer to the service provider is also 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 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 most recent research paradigm which is referred to as *Nomadic Mobile Services Provisioning*. The term *Nomadic Mobile Service* (NMS) is coined in [4]. An 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. As compared to the service provisioning in the fixed network, to develop, deploy, use and maintain an NMS, a variety of concerns need to be addressed because of the following reasons: intermittent

The 2007 International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST 2007)
0-7695-2878-3/07 $25.00 © 2007 IEEE

bandwidth characteristics and high error rate of the wireless communication link, reachability and connectivity issues arising because of mobility, performance concerns as a reason of the scarce computation resources of a mobile device as compared to their counterparts in the fixed network and lack of standardized protocols.

The early work in this direction has resulted in a few NMS provisioning approaches. For instance, in [4] van. Halteren et. al. propose a proxy based middleware based on the *Jini surrogate architecture* [5]. In [6], Srirama et. al. present a prototype implementation of mobile web service provisioning and the discovery of services in a *peer to peer* (P2P) network. Pratistha et. al. [7] proposes an asymmetric lightweight infrastructure referred to as *Micro-Services* capable of hosting web services from the mobile devices. From now on, we refer to them as NMS_{Proxy}, NMS_{P2P} and $NMS_{Assymmetric}$ approaches respectively. Each of these approaches employs a different architecture and address a different mix of issues. This paper provides a comparative study of these NMS provisioning approaches based on their architectural design, development choices and prototyped applications. This study will be helpful for the newcomers to get acquainted to this area and will help the application developers to choose the suitable approach which will serve the needs of the service and its clients.

This paper is structured as follows: Section 2 briefly introduces three NMS approaches and provides their architectural overview. Section 3 compares software and computing platform support for the development of NMS using these approaches. Section 4 presents the case studies of individual approaches. Section 5 summarizes the paper and discusses the future directions.

2. NMS Provisioning Architectures

The fundamental components of SOA include *basic services*, their *descriptions*, and basic operations such as *publication*, *discovery*, *selection*, and *binding* that produce or utilize such descriptions [8]. The components *service directory* and *registry/repository* coordinate the interaction between service and client. Based on the above, we compare the NMS provisioning approaches and discuss their merits and demerits using the following: a) service hosting on the mobile device; b) service publication and discovery and c) service usage. We give a special emphasis on the features which are used to handle nomadic computing aspects.

Before providing the comparison of these approaches, since NMS_{Proxy} uses *Jini* and NMS_{P2P} and $NMS_{Assymmetric}$ use *web services* as the underlying technologies, here we briefly discuss them, so as to understand the further sections.

The Jini system architecture consists of three categories: *programming model*, *infrastructure*, and *services* [9]. 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.

The basic architecture for web services model consists of the following five layers [10]: the *transport layer* exchanges request-response messages using protocols like HTTP(S). The *packaging layer* uses XML-based SOAP protocol to package the messages to be exchanged. The *information layer* carries the SOAP messages and provides functionality to encode and decode these messages. The *services layer* uses *Web Services Description Language* (WSDL) to describe a SOAP/XML Web service. The *discovery layer* publishes the information about web services and provides a mechanism to discover the available web services through the *Universal Description, Discovery and Integration* (UDDI) specification.

2.1. NMS Hosting

Figure 1 provides an overview of the NMS provisioning approaches under study. An NMS prototyped using NMS_{Proxy} approach 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.

Though NMS_{P2P} is built for resource constrained devices like cellular phones, it followed the same architecture as general web services. The architecture supports smart phone acting as a mobile web service (MWS) provider and also as a client. Thus each smart phone acts as a peer in the mobile P2P network [11].

$NMS_{Assymmetric}$ approach also follows the general web services architecture. However, the NMS

The 2007 International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST 2007)
0-7695-2878-3/07 $25.00 © 2007 IEEE

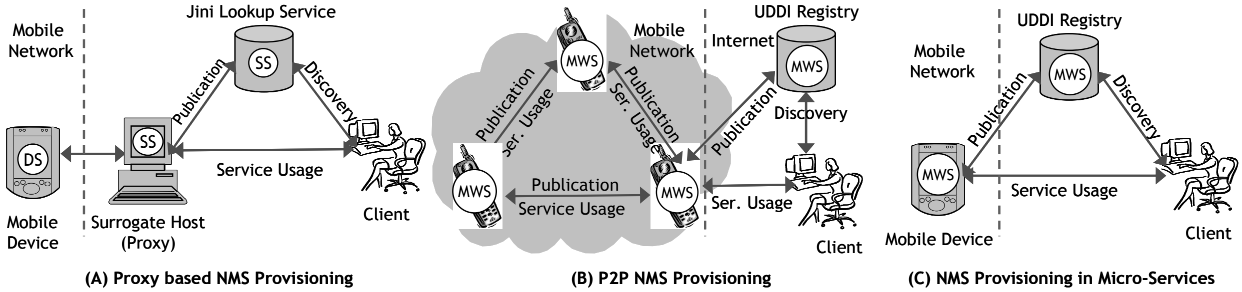

Figure 1: High Level Overview of the NMS Provisioning Architectures

prototyped using $NMS_{Assymmetric}$ approach accepts only SOAP simple types (e.g. *String, Integer, Char*) to avoid the complexity in performing the extraction process of some SOAP complex types. Since this feature is included to address the resource limitations of a mobile device, we refer to this approach as an *assymmetric approach* according to the classification in [12].

The NMS_{Proxy} approach has certain advantages over the other two approaches, because of its choice to use a surrogate to represent a device service. A surrogate may execute complex functions, thus off-loading a processing limited device. 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 an NMS 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. Further, an appropriate decision should be made to assign specific functionalities to the device service and surrogate.

With the NMS_{P2P} the smart phone becomes a multi-user device where the owner/carrier of the device can work in parallel with users of the NMS without explicit effort on his/her side. By following the general web services architecture, NMS_{P2P} makes the NMS interoperable. But unlike NMS_{Proxy} with a pure decentralized approach in the wireless network, providing reasonable QoS to the client is quite difficult.

For the NMS service hosting the $NMS_{Assymmetric}$ approach has similar merits and demerits as NMS_{P2P} approach as far as web services are concerned. However, the service capabilities remain limited due to the support for only SOAP simple types and moreover scalability is an issue because of the limitations on the wireless bandwidth and the processing capabilities of the mobile device (in case multiple clients connect to the same service).

2.2. NMS Publishing and Discovery

In the NMS_{Proxy} approach, the surrogate contacts the Jini lookup service for the service registration. After the lookup service is discovered either through *unicast* or *multicast* discovery [9], 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, then the lookup service will discard it.

As the $NMS_{Assymmetric}$ approach follows the general web services architecture, the NMS description in WSDL is published in a UDDI registry.

Since the NMS_{Proxy} and $NMS_{Assymmetric}$ approaches rely on the underlying service discovery mechanism i.e. Jini and Web Services respectively for the lookup service discovery by both, NMS and its consumer, they inherit the characteristics of these mechanisms. In both of these approaches, after discovering a lookup service a client requests a service by providing the description of the desired service. If the desired service is registered with the lookup service, the client receives the information to access the service (In Jini, the client downloads a service proxy which communicates with the actual service.). The client uses the service according to its service description.

NMS_{P2P} approach states that the traditional centralized UDDI based registries have many limitations and are not the best solution for NMS discovery, as they are susceptible to the single point of failure and cause the bottleneck. Besides, the node mobility makes the binding information in the WSDL documents inappropriate. Since the NMS are to be republished every time, this process leaves many stale advertisements in the registry. To overcome these difficulties, NMS_{P2P} relies on the P2P network for advertising and discovery of the mobile web services. The approach [13] was prototyped using the JXTA

The 2007 International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST 2007)
0-7695-2878-3/07 $25.00 © 2007 IEEE

network [14]. Here the WSDL description of NMS is incorporated into *Module Specification Advertisements* (MSA), so that the NMS is sensed as a JXTA service among other peers. Consequently, the discovery of the NMS is according to JXTA service discovery principles; thus possesses all the advantages of the P2P discovery. However the NMS_{P2P} approach does not block the service provider from publishing the NMS to the UDDI registry. As long as the mobile hosts are provided with public IP address and the services are published at UDDI, the NMS are accessible by any client from Internet.

2.3. Communication between NMS and Client

In NMS_{Proxy} approach, an NMS client communicates with the surrogate using any remote invocation protocol e.g. *Java Remote Method Invocation* (RMI). 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. For this purpose, NMS_{Proxy} uses an *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.

In NMS_{P2P} approach, a mobile device consists of a *Web Service Handler* built on top of the normal web server. An *HTTP Interface* listens for incoming client requests on a sever socket and transfers them to the component called as *Request Handler*. When the Request Handler determines that the request is a normal HTTP request, it processes the request just as a standard web server and sends the response to the client. If the message comprises a web service request; the Web Service Handler extracts the Java objects, using the *kSOAP2 Processor*. After that the objects are passed to the *Service Handler*, which extracts the service details (parameters) and invokes the respective service. The response message is then constructed and sent to the client by the Request Handler.

To handle client requests, the $NMS_{Assymmetric}$ approach provisions three components. The *Compact Listener* component receives client requests and also sends the response back to the client. The *Core Server* receives encapsulated HTTP requests from the Compact Listener. It then performs the necessary validations and determines the appropriate *Supporting Modules* to forward these requests to. Lastly, the Supporting Module represents the implementation of a particular Internet protocol e.g. HTML, SOAP [15].

NMS_{P2P} and $NMS_{Assymmetric}$ handle the client requests directly, hence it is expected that the response time from the mobile device to the client is relatively lesser as compared to NMS_{Proxy}. However, in case of the temporary disconnection, NMS_{Proxy} approach may offload a Device Service by caching/storing data at its Surrogate and reply the client request accordingly.

A major concern for NMS provisioning is to ensure the reachability of the service to the client. An NMS is not fixed and changes its location resulting in variable IP address assignments to the mobile device. While a few mobile operators dynamically assign global IP address to the mobile device, others assign a local IP address. In the later case, NAT inhibits connection from the public Internet to a host behind a NAT router. To solve this problem, NMS_{Proxy} piggybacks the messages from the surrogate (and in turn from the client) to a device service in an HTTP Response to the HTTP Request from the device which is periodically sent. To solve the reachability problem, NMS_{P2P} also proposes an alternate approach to handle the client requests. Here the web service requests are transmitted over the JXTA pipes. The approach eliminates the need for a public IP address for each mobile host; as within the P2P network, the services can now be addressed with the *PeerID* of the devices. The mapping between the unique *PeerID* and IP of the devices is taken care by the JXTA platform. In the literature ([7,15]), there is no information available regarding how $NMS_{Assymmetric}$ approach handles the reachability problem.

3. Software and Computing Platform Support

This section addresses the implementations of the various NMS approaches and the required platforms. For this purpose, we identify the necessary aspects as follows: 1) target mobile computing environments; 2) NMS provisioning software implementation; 3) service and client implementation. For the sake of brevity, we present such comparison in the tabular format and provide explainations whereever necessary.

Table 1 shows the target mobile devices, software requirements for the services and supported communication networks for the NMS provisioning approaches under study. Table 1 also shows the other aspects like programming platform, footprint, base technology and messaging support for the NMS provisioning software.

Table 1: NMS Development Choices

Feature	*NMS_{Proxy}*	*NMS_{P2P}*	*NMS_{Assym}*
Mobile Devices	PDAs, Smart Phones	Smart Phones	PDAs
Operating System	Windows Mobile, Pocket PC, Linux	Any OS supporting J2ME/CLDC	Windows Mobile, Pocket PC
Software Platform	J9 JVM, J2ME Personal Profile 1.0	kVM, J2ME MIDP 2.0	Microsoft .net CF
Base Technology	Jini Surrogate Architecture	Web Services, JXTA	Web Services
Comm. Networks	Any network supporting HTTP 1.1	Any network supporting HTTP 1.1 and assigning public IPA	Any network supporting HTTP
Program. language	Java JDK 1.3, J2ME	J2ME / Personal Java	Microsoft .net
Footprint	153KB on mobile device, 257KB including surrogate	J2ME version: 119KB, Personal Java version: 130KB	86KB
Messaging Support	HTTP GET, POST and streaming using HTTP Chunking	HTTP GET, POST and SOAP over HTTP	HTTP GET, POST and SOAP over HTTP
NMS Elements	Device service, surrogate object, service proxy & interface	Service, WSDL description	Service, WSDL description
NMS-Client Communic-ation	Any RPC middleware e.g. Java RMI, Jini Remote Eventing may be used to notify client	SOAP over HTTP	SOAP over HTTP

In addition to the basic NMS provisioning, NMS_{Proxy} middleware also features context-awareness by exploiting multi-homing feature of the mobile device. This features provides the capability of selecting the network interface which offers higher bandwidth among the available network interfaces providing internet connectivity [16]. For this purpose NMS_{Proxy} uses an external context source COSPHERE [17]. COSPHERE footprint is 90 KB and runs on the mobile devices having *Windows Mobile 2002* as their OS. An introductory tutorial for the development of NMS using NMS_{Proxy} approach is available at [18].

NMS_{Proxy} approach relies on Jini and in turn on Java for the development, which is in contrast to the other two approaches which follow the general web services architecture to make the NMS interoperable between the development environments.

4. Existing NMS Applications

This section provides a brief overview of the applications developed using each NMS provisioning approaches under study. The purpose of this section is to show that NMSes are already in use and receiving acceptance in domains like *m-health*, *entertainment*, *context-aware computing* and *mobile business*.

4.1. NMS_{Proxy} Applications

The *Remote Patient Tele-Monitoring Service* [4] gathers patient's data collected from medical sensors attached on the patient's body and delivers this data in a near real-time fashion to the healthcare professionals. The *Tele-monitoring Device Service* (TDS) communicates with the sensor system using *Bluetooth* connectivity, collects the vital signs and streams them to the *Tele-monitoring Surrogate* (TS). The client located at the health-care center displays these vital signs graphically for the use by the health-care professionals. The *Nomadic Positioning Service* [19] provides the current position of a mobile device for use by a context aware application interested in this information. The *Positioning Device Service* (PDS) uses *Place Lab* library that determines a position of the mobile host by spotting beacons, such as Wi-Fi access points. Whenever PDS detects a change in the location of mobile host, it sends the location change event to the *Positioning Surrogate* (PS), which is later sent to the interested clients. The *Nomadic Robot Service* [20] provides the capability to a client to control the robot as desired. The *Robot Device Service* (RDS) communicates with the *LEGO Mindstorm robot* using *Infrared* connectivity. The RDS receives instructions from the client via *Robot Surrogate* (RS) and instructs the robot to move accordingly. Another application in the area of context-aware computing uses the concept of NMS to realize context sources on the mobile device as services and make this context information available to the *Context Distribution Framework* in the Internet [21].

The 2007 International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST 2007)
0-7695-2878-3/07 $25.00 © 2007 IEEE

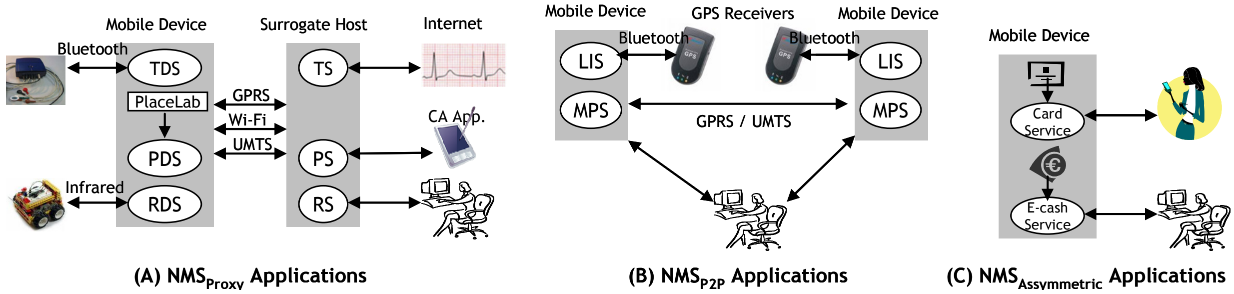

Figure 2: NMS Applications in Diverse Domains

4.2. NMS_{P2P} Applications

The basic elements of most of the services developed by NMS_{P2P} are the *Location Information Service* (LIS), and the *Mobile Picture Service* (MPS). The LIS communicates with an external GPS receiver using Bluetooth and provides the GPS coordinates of the device. The MPS transfers the pictures taken by the smart phone on the fly. These services provide a building block for various applications in the domains of *m-learning*, *social networks* and *computer supported co-operative work environments*. For example, the *collaborative journalism application* involves the smooth co-ordination between journalists and the news editors. Journalists cover different events like sport events, conferences etc. across the globe and an editor keeps track of the location of her journalists and the content they have provided. A journalist uses the services of other members to synchronize their activities. In *m-learning* application; scenarios like *podcasting*, *mobile blogging*, *mobile learning media sharing service* and *expertise finder service* are currently being studied. The major commercial aspect with NMS_{P2P} is the possibility for small mobile operators to set up their own mobile web service business without resorting to stationary office structures, thus going one step further in the move from central to P2P architectures.

4.3. $NMS_{Assymmetric}$ Applications

The two proof-of-concept NMSes developed using $NMS_{Assymmetric}$ approach are the *Business Card Exchange Service* and *e-Cash Transfer Service*. The business card and e-Cash tokens are represented as String type in the SOAP encoding. On receiving a business card from the other mobile device, the client acknowledges the submitted information as being received and adds the card details to the contacts database in the *PocketPC Outlook* program.

5. Summary and Future Directions

Nomadic Mobile Services are a result of the advances in the mobile devices and communication technology combined with the latest developments in the area of SOA. By using various NMS provisioning approaches, applications are prototyped successfully in the areas such as m-health, collaborative learning, entertainment, context-aware computing, location based computing and personal information exchange. This paper provides a comparative study of three NMS provisioning approaches based on their architectural design, development choices and prototyped applications.

Among the three approaches studied, NMS_{Proxy} suffers from interoperability, NMS_{P2P} suffers from the scalability and $NMS_{Assymmetric}$ suffers from the limited service capabilities. To address these issues, a solution which uses web services for better interoperability, and employes proxy approach for the better QoS could be a possible technical design. Current research of NMS_{P2P} is working on some of these issues with an *Enterprize Service Bus* [22] *based mediation framework*, handling QoS of the mobile hosts [23].

Despite the successful applications of NMS, the current state of research in this field is far from complete. There is a necessity to research the processes for NMS development, deployment and maintenance. Some technical issues such as service reachability may find a solution in the near future e.g. by the widespread acceptance of mobile IPV6. However, there is still a need to assess the potential of NMS for the revenue generation, social interaction and widespread acceptance. Furthermore, though there are standardization activities going on for the use of mobile devices as web services client e.g. OMA [24], there is also a need to initiate the standardization activities for NMS provisioning as well.

0-7695-2878-3/07 $25.00 © 2007 IEEE

Acknowledgement

This work is supported by Freeband Awareness project (under grant BSIK5902390), Amigo project (IST-004182, partially funded by the European Commission) and Research Cluster Ultra High-Speed Mobile Information and Communication (UMIC) (http://www.umic.rwth-aachen.de/).

References

[1] Dokovski, N., Widya I., and Halteren, A.v., *Paradigm: Service Oriented Computing*. Freeband/AWARENESS /D2.7b, http://awareness.freeband.nl, 2004.

[2] Calder B. et. al., *JSR 172: J2ME Web Services Specification*. Java community process, http://jcp.org/en /jsr/detail?id=172, 2004.

[3] Balani, N., *Deliver Web Services to mobile apps*. IBM Developer Works, http://www-128.ibm.com/develo perworks/wireless/edu/wi-dw-wiwsvs-i.html, 2003.

[4] Halteren, A.v. and Pawar P., *Mobile Service Platform: A Middleware for Nomadic Mobile Service Provisioning*. 2nd IEEE International Conference On Wireless and Mobile Computing, Networking and Communications (WiMob 2006), Montreal, Canada, June 2006.

[5] Sun Microsystems, *Jini Technology Surrogate Architecture Specification*. https://surrogate.dev.java.net /doc/sa.pdf, October 2003.

[6] Srirama, S.N., Jarke, M. and Prinz, W., *Mobile Web Service Provisioning*. International Conference on Internet and Web Applications and Services (ICIW'06), Guadeloupe, French Caribbean, 2006.

[7] Pratistha, D., Nicoloudis, N. and Cuce, S., *A Micro-Services Framework on Mobile Devices*. International Conference on Web Services, Nevada, USA, 2003.

[8] Papazoglou, M.P. and Georgakopoulos, D., *Service Oriented Computing*. Communications of the ACM, 46(10): p. 24-28, 2003.

[9] Sun Microsystems, *Jini Architecture Specification*. http://www.sun.com/software/jini/specs/jini1.2html/jini-title.html, 2001.

[10] Engelen, R.v., *Code Generation Techniques for Developing Light-Weight XML Web Services for Embedded Devices*. 19th Annual ACM Symposium on Applied Computing (SAC 2004), Nicosia, Cyprus, 2004.

[11] Shirky, C., et al., *P2P networking overview*. O'Reilly, Sebastopol, California, 2001.

[12] La Porta, T. F., Sabnani, K. K., Gitlin, R. D., *Challenges for Nomadic Computing: Mobility Management and Wireless Communications*. Mobile Networks and Applications, 1(1): p. 3-16, 1996.

[13] Srirama, S. *Publishing and Discovery of Mobile Web Services in Peer to Peer Networks*. International Workshop on Mobile Services and Personalized Environments (MSPE'06), Aachen, Germany, 2006.

[14] Gong, L., *JXTA: A network programming environment.* IEEE Internet Computing, 5(3): p. 88-95, 2001.

[15] Pratistha, D., *Exposing Web Services on Mobile Devices*, school of Computer Science and Software Engineering, Monash University, Australia, 2002.

[16] Pawar, P., Beijnum, B. J. v., Peddemors, A., Halteren A. v., *Context-Aware Middleware Support for the Nomadic Mobile Services on Multi-homed Handheld Mobile Devices*. To be presented in 12th IEEE Symposium on Computers and Communications (ISCC 2007), Aveiro, Portugal, 2007.

[17] Peddemors, A., Eertink, H., Niemegeers, I., *Communication Context for Adaptive Mobile Applications,* 3rd Conference on Pervasive Computing - Middleware Support for Pervasive Computing Workshop (PerWare'05), New York, USA, March 2005.

[18] *Mobile Service Platform,* http://janus.cs.utwente.nl: 8000/twiki/bin/view/MSP/Developers, 2006.

[19] Uiterkamp, E.S., *Nomadic Positioning Services for a Mobile Service Platform*, ASNA Group, Department of Computer Science, University of Twente, The Netherlands, 2005.

[20] Tol, P.v., *Service discovery of Lego Mindstorms based nomadic services*, ASNA Group, Department of Computer Science, University of Twente, The Netherlands, 2005.

[21] Pawar, P., Halteren, A.v., Sheikh, K., *Enabling Context-Aware Computing for the Nomadic Mobile User: A Service Oriented and Quality Driven Approach*. IEEE Wireless Communications & Networking Conference (WCNC 2007), Hong Kong, 2007.

[22] Schulte, R.W., Predicts 2003: Enterprise service buses emerge. Gartner Report, 2002.

[23] Srirama, S.N., Jarke, M., Prinz, W., *A Mediation Framework for Mobile Web Service Provisioning*. 10th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW'06), Hong Kong, 2006.

[24] Open mobile alliance group, *Mobile Web Services Requirements - Version 1.0*, http://www.openmobile alliance.org/release_program/owser_v10.html, 2003.