Mobile Web Services in Health Care and Sensor Networks

Fahad Aijaz, Muzzamil Aziz Chaudhary, Bernhard Walke
Department of Communication Networks
Faculty 6, RWTH Aachen University
Kopernikusstr. 5, 52074 Aachen, Germany
Email: {fah, muz}@comnets.rwth-aachen.de

***Abstract*—The Wireless Sensor Networks (WSN) environments are increasingly becoming intelligent with the rapid boost in the capabilities of sensor nodes. The Sun Small Programmable Object Technology (SPOT) platform offers a high-end sensing device in terms of processing, memory and battery configurations compared to other existing in the community.**

Within the scope of this paper, a health care use case is derived from the concept of service-oriented collector nodes represented by Sun SPOTs. The existing Mobile Web Services (MobWS) framework is ported to Sun SPOT devices that enables domain specific in-network computations enfolding synchronous or asynchronous classes of MobWSs. Based on that, the exemplary requirements of the Web Services (WS) in the health care domain are spotlighted and impressions of a comprehensive prototype, called MEDICARE, are shown.

I. Introduction

In principle, every ubiquitous context-aware application demands a technology platform capable of sensing, processing and storing environmental information. These platforms are usually represented by small devices equipped with sensors, a microprocessor and some kind of volatile or non-volatile memory. The networking capabilities of these devices vary across application use cases and may employ wired, wireless or both modes of communication. Here, the focus is particularly on the wireless mode. The small-footprint applications are developed on top of these devices which uses their networking capabilities to communicate with other similar nodes, or even with external high-end servers or network terminals. For such platforms, enfolding wireless communication, several interfaces have been standardized by the IEEE, which includes 802.15.1 (Bluetooth), 802.11 (WLAN), 802.16 (WiMax), 802.20 (UWB) and 802.15.4 (LR-WPAN). Among these wireless communication interfaces, the IEEE 802.15.4 is the underlying standard for the ZigBee[1] protocol. Each of these standards differs in terms of their characteristics, such as range, reliability, throughput and cost. However, one of the most influential factor in selecting the right standard for limited configuration device platforms is the power consumption. Especially for applications demanding mobility of devices, the importance of power consumption cannot be ignored. Thus, the devices capable of obtaining power from other resources, like batteries or solar cells, are used to increase mobility and decrease the dependency on wired power supply. For these devices, correctly choosing the wireless interface standard becomes even more critical.

[1]http://www.zigbee.org

II. Platforms for Wireless Sensor Networks

In the past, extensive research efforts have been put to innovate technology platforms for ubiquitous context-aware computing. The Mote platform [1], [2] is recognized as the most dominating, battery powered platform, for data acquisition in the research community [3]. Related platforms, with similar characteristics as Mote, have also been developed by other research groups [4]–[6]. However, the limited storage capacity of the Mote prevents the realization of complex algorithms [7] which might be helpful in optimizing the performance, e.g. by reducing the data transmission frequency. Such optimizations directly influence the power consumption characteristics of the device. The programming paradigm, nesC [8], used for the Mote based development is an extension of the C programming language, and fundamentally based on the execution model of the TinyOS[2]. Compared to other high-level languages, such as Java, the programming in the nesC is complex and, like the C programming, lacks explicit exception handling mechanisms. The profiling and debugging facilities of the embedded application on the devices during the execution mode are also limited. Majority of the existing technology platforms employ the C or the assembler as the programming paradigm that prolongs the learning-curve, by imposing the need of hardware-level knowledge for the application developer familiar only with the high-level programming languages. This makes the configuration and the deployment of the applications, network and devices, a cost inefficient and lengthy process.

A. The Sun SPOT Platform

In order to address the problems imposed on the software development by the existing devices [3], a project is originated at the Sun Microsystems Laboratories[3] that envisage the construction of a sensor-equipped high-valued device to support ubiquitous computing applications. According to a technical report published by Sun [3], the project mainly focuses on providing efficient solutions to some major issues faced by existing technology platforms, such as:

[2]http://www.tinyos.net
[3]http://research.sun.com/

- The construction of a general-purpose device to support data acquisition applications and control systems.
- Facilitate the implementation and productivity of complex applications, by providing easy-to-use development tools, based on wireless sensor platforms in order to meet the commercial needs.
- Provide a high-valued device in terms of processing and memory capabilities than those currently existing.
- Develop support for dynamic Over-The-Air (OTA) deployment in order to ease the software configuration process.

As of now, the project has resulted in a device called the Sun SPOT[4] (see Figure 1) that provides an innovative solution to the aforesaid issues in terms of the software and the hardware platforms.

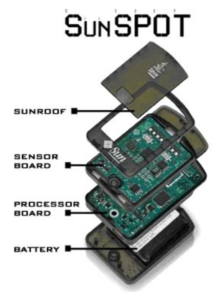

Fig. 1. The Sun SPOT Device from Sun Microsystems

1) The Software Platform: The aim of the software platform is to enable portability, reduce the development time and cost while easing the deployment process for the WSN applications. Thus, the Sun SPOT software platform enfolds Java Micro Edition (Java ME) as the fundamental programming language enabling cross-platform portability, due to the platform independence, and the ease of development based on the variety of standard development tools already known within the programmers' communities. In the past, experience has shown that the productivity of a software developer increases four times with Java [9], [10] due to the abstraction it provides over many low-level tasks that are required by conventional programming languages, such as the C. Therefore, for ensuring the all-in-Java philosophy, an extremely light-weight Java Virtual Machine (JVM) called Sqwawk [11], [12] is implemented and the Connected Limited Device Configuration (CLDC) of the Java ME technology has been chosen. The software platform of the Sun SPOT devices comes with the complete Software Development Kit (SDK) that offers a comprehensive set of the Java libraries for the developers. Using the Sun SPOT SDK, the development of the low-level sensor-based context-aware applications has taken the same form as any other higher-layer Java application, by facilitating hardware access through the standard libraries without demanding any prior knowledge about it. The technical report [3] provides a coding example, by comparing the Java and the nesC, for controlling the hardware components, such as LEDs.

[4]http://www.sunspotworld.com

2) The Hardware Platform: The hardware platform of the Sun SPOT node complements its software configurations by providing 180 MHz 32 bit ARM920T core processor, 512K RAM/4M Flash, 2.4 GHz radio with integrated antenna, 3.7V rechargeable 720 mAh lithium-ion battery, IEEE 802.15.4 based radio communication and Public Key Cryptography (Elliptical Curve Cryptography (ECC)). The device offers a sensor board with three kinds of sensors, 2G/6G 3-axis accelerometer, temperature and light, along with the 8 tri-color LEDs and two momentary switches for attaching additional sensors. The battery issues are addressed by defining three power consumption modes namely the Run, the Idle and the Deep-sleep [13]. The execution lifetime of a Sun SPOT ranges from minimum of ≈ 6 hours (by consuming maximum of 120 mA current in Run mode) to maximum of ≈ 2.6 years in Deep-sleep mode with 32 μA consumption of current.

The project Sun SPOT provides a simple and flexible sensor-oriented hardware and software platform for experimenting pervasive applications, which also addresses the shortcomings of the battery powered WSN platforms. The feedback received from early beta testers is promising and shows that the device supports the ease of application creation, high memory and processing capabilities than existing sensor platforms, and the OTA deployment [3].

III. Service-oriented Wireless Sensor Networks

Service oriented computing, based on Service Oriented Architecture (SOA) [14], is a widely practiced technique for integrating heterogeneous applications. Within the scope of this work, we apply the principles of SOA to WSN in order to facilitate the high-valued context-aware application use cases based on MobWS.

With the significant advancements in the WSN research, these networks are not just the data networks with the sensors being the sources of data. The WSN are now deployed with a goal to fulfill specialized application and environmental needs. Since in such scenarios, the large number of sensors/actuators are distributed across the network infrastructure, it is obvious for the sensing node to receive huge amount of raw data from its environment. Typically, the sensor nodes responsible for collecting and distributing environmental data have limited capabilities to store and process this information. A recent literature [15] addresses these shortcomings by introducing a concept of collector nodes, which are better equipped with processing, storage and battery capabilities compared to the ordinary sensors. These collector nodes receive the data from the spatially distributed sensors and perform complex in-network computations according to the application needs. Consequently, the outcome of the computations can be presented to some higher hierarchy in the network.

In our recent research [16], we proposed to deploy MobWS on these collector nodes to perform the distributed and Peer-to-Peer (P2P) in-network computations. Since the in-network computations in the wireless networks is a scientific challenge, the deployment of MobWS framework [17], [18] helps in bringing the server capabilities to the collector nodes, consequently, supporting the highly complex MobWS provisioning from the resource-constraint devices to perform the collaborative distributed operations. As an extension to [16], here we use the service-oriented WSN to derive a specialized use case focusing on the exemplary health care scenario.

IV. Mobile Web Services in Health Care

The demands of health care solutions are not straightforward. Especially in context-aware applications based on WSN infrastructure, the distributed collector nodes, while collecting sensor data, are required to perform complex processing simultaneously to deliver meaningful information. The computation processes performed by these collector nodes are classified as short- or long-lived. We further map each respective class of process to Mobile Synchronous Web Service (MSW) and Mobile Asynchronous Web Service (MAW). In order to clearly specify the requirements, the MobWSs in health care scenario are categorized as Environmental, Patient and Equipment sensing services. Each category may incorporate several MobWSs offering interfaces for short-lived or long-lived processing. Table I lists some exemplary services in each category that may be provisioned by the collector nodes to fulfill the requirements of the health care use case.

Category	Services	Type	
		short-lived	*long-lived*
Environmental	LightCondition	x	
	RoomTemparature	x	
	NoiseDetection		x
	Presence		x
	DoorStatus	x	
Patient	BodyTemparature		x
	BloodPressure	x	
	HeartCondition		x
	PulseSignals		x
	Movement		x
Equipment	DeviceConfiguration	x	
	ModeSelection	x	
	PerformanceMonitor		x
	ValvesCondition	x	
	OxygenMonitor		x

TABLE I
Categories and Example Services in Health Care

A. Architectural Requirements

Developing a comprehensive health care solution demands an architecture incorporating both MSW and MAW services. Each type of service has its own requirements which are strictly based on the specification of use case. Here, the service examples are presented to spotlight the importance and necessity of each class of service in software architectures of health care applications, which may differ from the actual implementations.

1) Mobile Synchronous Web Services: The short-lived services perform operations that are instantaneous request-response processes. These services are developed as MSW. In Table I, services that perform small unit of task, such as fetching the data or checking the status, are shown as short-lived under each category. For instance, the Environmental services are responsible for sensing information in the patient's surroundings. Services like LightCondition, RoomTemparature and DoorStatus may not require continuous information sensing capabilities. These services are invoked to instantly fetch the current environmental information that answers simple questions like; What is the current room temperature? Are the lights switched on? or Is the door open? The outcome generated by these services is usually a simple value that can be calculated, using the sensor data, within a very short span of time. The short-lived services can also be used to poll the information after pre- or user-defined intervals. Furthermore, the Patient category deals with services strictly related to the health conditions of a patient. In Table I, the BloodPressure service is presented as a short-lived process that may be invoked to get the blood pressure readings periodically. Similarly, the Equipment category lists services that are related to the medical equipment, like a ventilator or Electrocardiography (ECG) devices, in patient's room that is used for the treatment. Under this category, the DeviceConfiguration, ModeSelection and ValvesCondition services in the current example are represented as short-lived services.

2) Mobile Asynchronous Web Services: On the contrary, long-live services are time consuming and continuous processes that may involve variable interaction techniques including request-response, solicit-response, one-way or notification. These interactions are derived from the standardized operations of Web Service Description Language (WSDL) [19]. The long-lived services are developed as MAW. The Environmental category in Table I shows NoiseDetection and Presence services as long-lived processes. The NoiseDetection service is responsible for continuously monitoring the patient's environment to prevent high-pitched sounds, such as produced by a loudly speaking visitor, dragging of furniture or even by a malfunctioning medical equipment. The requirements of NoiseDetection service demand continuous sensing of environment, however the alerts generated by the high-pitched signals may only be transmitted when required, e.g. if a predefined noise threshold is crossed. The Presence service, on the other hand, may be realized to detect the existence of people in patient's room. This can help in preventing crowd formation and may require continuous execution as a long-running process. The Patient in Table I is the most suitable category for long-lived services. In most cases, the health conditions of a patient require continuous monitoring to analyze the current status and predict forthcoming circumstances. For instance, the BodyTemparature, HeartCondition, PulseSignals and Movement services may be implemented as long-lived processes to continuously sense and disseminate the health related readings of a patient. In the same way, PerformanceMonitor and OxygenMonitor services under the Equipment category may require continuous execution as they are likely to influence the health condition of the person under observation. The long-lived services do not only perform continuous processes, but they also support runtime management in terms of controlling a service by changing its state or configuration. For instance, the heartbeat alert threshold in the HeartCondition, or oxygen level threshold in the OxygenMonitor service, may be adjusted automatically or manually, during the execution phase based on the analysis of previous readings.

V. Service Provisioing Sensor Nodes

Considering the fundamental requirements specified earlier, a proof-of-concept, called MEDICARE, has been developed to demonstrate a health care use case by bridging the gap between service-oriented computing and WSN domains. In the first phase, the MobWS framework [17], [18] is ported to the collector nodes, represented by Sun SPOTs, consequently equipping them with MSW and MAW provisioning capabilities. The second phase shows some impressions of MEDICARE, the management and visualization application, for high-end network servers to monitor and control Sun SPOT nodes and the embedded MobWSs.

Porting the MobWS framework to Sun SPOT devices is not straightforward. Since MobWSs are not limited to consumer devices, therefore they can be used as back-end or internal processes of network terminals with no direct interaction with the end user. In health care application scenario, these network terminals are the Sun SPOT devices forming a distributed adhoc WSN. From the technological perspective, the Information Module Profile (IMP) is a standard way to develop Java ME applications for embedded devices. Thus, as a prerequisite for porting the framework to Sun SPOTs, a transition from Mobile Information Device Profile (MIDP) to IMP compatible is made. In the following, a protocol binding extension for the framework to fulfill communication requirements of Sun SPOTs is discussed.

A. SOAP over IEEE 802.15.4

The MobWS framework was originally developed to communicate over variety of communication protocols [20]. Due to the transport neutral behavior, Simple Object Access Protocol (SOAP) [21] was utilized as an underlying standard messaging framework for MobWSs. However, the protocol bindings developed in the original work focused on long-ranged communication using mobile phones [18], [22]–[24]. Since the Sun SPOTs communicate over the IEEE 802.15.4 radio standard, therefore a protocol binding for SOAP is developed to enable P2P short-ranged MobWSs.

The Generic Connection Framework (GCF) of Java ME provide connectivity libraries to communicate over the variety of protocols. The Sun SPOT SDK extends the GCF with two additional protocols, *Radiostream* and *Radiogram*, to support short-ranged radio multi-hop communication between the nodes [25].

1) Radiostream Protocol: The radiostream is a reliable protocol designed for socked-like P2P communication and supports buffered stream-based IO between mobile nodes. From the software developer's point of view, the connection establishment process, similar to sockets, requires a 64 bit IEEE Address and a Port Number within the range of 32-255. In Sun SPOT environment, ports from 0-31 are reserved for system use. The radiostream protocol does not support automatic handshaking during the connection establishment phase, however every data transmission between the mobile nodes supports an internal Medium Access Control (MAC) level Acknowledgment (ACK) mechanism. Due to the multi-hop nature, the final destination node of the transmitted radio packet dispatches an ACK to the sender, who resides in a blocked state, if and only if the destination node is reached at the next hop. On the contrary, if the next hop is not the final destination node, the ACK is delivered asynchronously while the sender does not remain blocked.

2) Radiogram Protocol: The radiogram is an unreliable class of the client-server protocol supporting datagram-based communication between the mobile nodes. The radio packets transmitted over this protocols may be lost, arrive multiple times or out of sequence. The connection establishments process from the developer's perspective is similar to the radiostream protocol and requires a 64 bit IEEE Address and Port Number within the same range specified earlier, while the radio packets are transmitted in the form of datagrams. The performance of the radiogram has been specially optimized for single hop connections. Due to its significant similarities with the User Datagram Protocol (UDP) [26], the radiogram supports unreliable multi-hop broadcasting functionality within the same Personal Area Network (PAN). The current Sun SPOT release does not provide inter-PAN broadcasting. The IEEE 802.15.4 radio standard enfolds the radiostream and radiogram protocols. Consequently, the SOAP binding to the standard incorporates message transmission over both protocols enabling reliable and unreliable forms of the short-ranged communication. The session management capabilities are derived from the previous work [22], [23] by employing WS Addressing [27]. A report in [25] provides conceptual and implementation level details of Sun SPOT communication based on IEEE 802.15.4.

B. Mobile Web Services on Sun SPOTs

Subsequent to the IEEE 802.15.4 binding for SOAP and framework transformation to IMP, MobWSs are developed for Sun SPOT nodes to enable flexible computation and communication provisioning capabilities. Since the applications developed on top of service-oriented WSN must be capable of performing short- and long-lived operations in order to completely meet the requirements, both classes of MobWS are implemented and deployed on the nodes with specialized focus.

The discovery of MobWSs on Sun SPOT nodes is a vital issue. Thus, each node must be able to provide information about its provisioned services to the peers and high-end network servers on demand. The MSWs, due to their short-lived nature, are suitable for such operations. Therefore, during the implementation phase, a MSW called ServiceList is developed that maintains a list of all the MobWSs published by its host Sun SPOT. This facilitates other Sun SPOT nodes or network servers to discover the provisioned services of their peers by invoking ServiceList and in response obtain the list of MobWSs available on that particular node. Once the services are discovered, MSW or MAW can be invoked to initiate short- or long-lived collaborative computations.

Within the current prototype, Sun SPOT nodes are not extended with third party sensors. Since the focus is to demonstrate the concept of health care solution based on MobWS-oriented WSN and in-network computations, the default Sun SPOT sensors, Light, Temperature and Movement, are used to mimic health care data. For each sensor, a MAW is implemented that continuously collects the environmental data and perform computations, such as calculating the Mean of every 100 values. In real situations, the computations performed by these services are derived from application requirements that may implement highly complex algorithms (e.g. media processing, data compression, face detection etc.). Here, the focus is to provide a framework and an Application Programmers Interface (API) that supports the development of domain specific MobWSs, of variable kinds, offering specialized computations in WSN. The Mean values calculated by the MAWcan be transmitted to the requesting node periodically or on demand. In the current prototype, computation outcomes are dispatched to the requesting clients after every 5 seconds that can be used to monitor the health condition and initiate respected actions. Due to the underlying architecture [28], [29], the prototype provides control, to the requesting node, over the behavior of MAWs by supporting state transitions during service runtime (e.g. suspending the computations for 30 minutes or changing the transmission interval to 10 seconds etc.).

C. MEDICARE Prototype Impressions

The collector nodes, represented by the Sun SPOTs, tightly couples with a high-end application that is responsible for the analysis and the representation of the computation outcomes to some higher hierarchy in the network. The application facilitates the MobWS creation, the invocation and management from a network server and may be extended to initiate domain specific processes based on the obtained computation results. The application is named MEDICARE and is based on the requirements specified in section IV. Some impressions of the MEDICARE prototype are presented in the Figure 2.

One of the limitations of the MEDICARE is the communication range. Since the Sun SPOT nodes support IEEE 802.15.4, it is required that the network server hosting the MEDICARE lies within the communication range of the nodes. Although the underlying standard is capable of multi-hop communication, this does not provide any solution for remotely accessing the Sun SPOTs. Thus, the prototype provides the integrated Web Server capabilities enabling the remote access to the MEDICARE over the Internet. The button with the globe icon shown on the top-right of all presented screen shots, transparently starts the MEDICARE Web Server in order to listen to the remote requests that changes the application status 'Online' as depicted in the Figure 2(c). From the implementation perspective, a standalone Java Applet version of the application is developed that connects to the MEDICARE Web Server in order to access the Sun SPOT nodes and the MobWSs. In this case, the MEDICARE on the network server relays all the requests from the Applet to the corresponding sensor nodes. A basic gateway functionality is also implemented that performs message transformations between the Sun SPOTs and the Applet. Thus, the complete functionality of the prototype is remotely accessible even from outside the communication range of Sun SPOTs, as illustrated by the scenario in Figure 3.

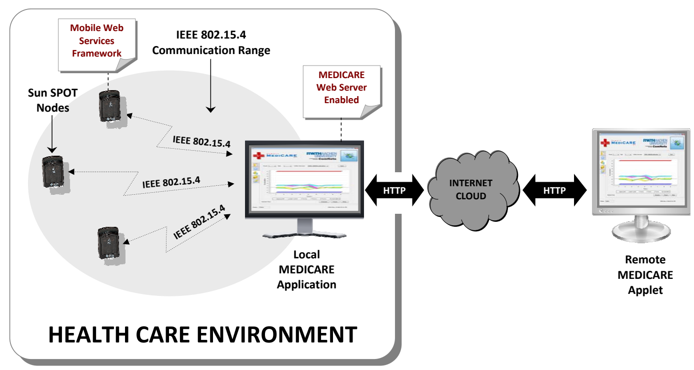

Fig. 3. MEDICARE Remote Access Scenario

VI. Conclusion

In this paper, a concept of MobWS-equipped collector nodes is used to derive a health care use case enabling service-oriented WSN. In the initial phase, a study is presented that compares the feasibility of Sun SPOT technology with several existing platforms in terms of ease of development, deployment and device configurations.

The research proposes to deploy MobWS on collector nodes, represented by Sun SPOTs, to perform in-network computations fulfilling the health care requirements. With the strong focus on health care sector, the paper presents two distinct interaction classes of MobWS and specifies their use within the target domain. In the later phase, a prototypical proof-of-concept called MEDICARE is presented and some impressions are shown. To enable MobWS provisioning Sun SPOT nodes, a reliable and unreliable SOAP binding for IEEE 802.15.4 is also developed.

References

[1] B. Warneke, M. Last, B. Liebowitz, and K. Pister, "Smart dust: communicating with a cubic-millimeter computer," *Computer*, vol. 34, no. 1, pp. 44–51, Jan 2001.

[2] B. Warneke, B. Atwood, and K. Pister, "Smart dust mote forerunners," 2001, pp. 357–360.

[3] B. Horan, B. Bush, J. Nolan, and D. Cleal, "A platform for wireless networked transducers," Sun Microsystems, Inc., Tech. Rep. SMLI TR-2007-172, September 2007. [Online]. Available: http://research.sun.com/techrep/2007/abstract-172.html

[4] M. Beigl, T. Zimmer, A. Krohn, C. Decker, and P. Robinson, "Smart-its - communication and sensing technology for ubicomp environments," Telecooperation Office (TecO), University of Karlsruhe, Tech. Rep. ISSN 1432-7864 2003/2, 2003. [Online]. Available: http://www.teco.edu/~michael/publication/tech_r1.pdf

(a) Sun SPOT Setup (b) Control Panel (c) Service Management

Fig. 2. Impressions of the MEDICARE Application

[5] E. Tuulari and A. Ylisaukko-oja, "Soapbox: A platform for ubiquitous computing research and applications," in *Pervasive '02: Proceedings of the First International Conference on Pervasive Computing*. London, UK: Springer-Verlag, 2002, pp. 125–138.

[6] A. Schmidt, "Ubiquitous computing computing in context," Ph.D. dissertation, Computing Department, Lancaster University, Computing Department, Engineering Building, Room A13, Lancaster University, Lancaster, England, U.K. LA1 4YR, November 2002. [Online]. Available: http://www.comp.lancs.ac.uk/∼albrecht/phd/

[7] K. Martinez, J. Hart, and R. Ong, "Environmental sensor networks," *Computer*, vol. 37, no. 8, pp. 50–56, Aug. 2004.

[8] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler, "The nesc language: A holistic approach to networked embedded systems," in *Proceedings of Programming Language Design and Implementation (PLDI)*, June 2003. [Online]. Available: http://nescc.sourceforge.net/papers/nesc-pldi-2003.pdf

[9] E. Christiansen and C. Quinn, "Java pays-positively," *IDC Bulletin*, vol. 16212.

[10] R. Wells, "Java offers increased productivity," Internet, June 1999. [Online]. Available: http://www.wellscs.com/robert/java/productivity.htm

[11] N. Shaylor, D. N. Simon, and W. R. Bush, "A java virtual machine architecture for very small devices," in *Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool support for embedded systems*, vol. 38, no. 7. New York, NY, USA: ACM, 2003, pp. 34–41.

[12] D. Simon, C. Cifuentes, D. Cleal, J. Daniels, and D. White, "Java (tm); on the bare metal of wireless sensor devices: the squawk java virtual machine," in *VEE '06: Proceedings of the 2nd international conference on Virtual execution environments*. New York, NY, USA: ACM, 2006, pp. 78–88.

[13] SunLabs, *Sun Small Programmable Object Technology (Sun SPOT) Theory of Operation*, part no. 820-1248-10, revision 1.0.8, may, 2007 ed., Sun Microsystems, Inc., 4150 Network Circle Santa Clara, CA 95045 U.S.A. 650 960-1300, May 2007. [Online]. Available: http://www.sunspotworld.com/docs/Purple/SunSPOT-TheoryOfOperation.pdf

[14] C. M. MacKenzie, K. Laskey, F. McCabe, P. Brown, and R. Metz, "Reference Model for Service Oriented Architectures," Published on the internet, Dec. 2005, oASIS Working Draft. [Online]. Available: http://www.oasis-open.org

[15] A. Giridhar and P. Kumar, "Toward a theory of in-network computation in wireless sensor networks," *Communications Magazine, IEEE*, vol. 44, no. 4, pp. 98–107, April 2006.

[16] F. Aijaz, S. M. Adeli, and B. Walke, "A service-oriented approach for in-network computations in wireless networks," in *In Proceedings of the Sixth IEEE and IFIP International Conference on Wireless and Optical Communications Networks*. Cairo, Egypt: IEEE / IFIP, April 2009, p. 6.

[17] F. Aijaz, B. Hameed, and B. Walke, "Asynchronous mobile web services: Concept and architecture," in *Proceedings of 2008 IEEE 8th International Conference on Computer and Information Technology*. Sydney, Australia: Internet, Jul 2008, p. 6. [Online]. Available: http://www.comnets.rwth-aachen.de

[18] G. Gehlen and L. Pham, "Realization and Performance Analysis of a SOAP Server for Mobile Devices," in *Proceedings of the 11th European Wireless Confernce 2005*, vol. 2. Nicosia, Cyprus: VDE Verlag, Apr 2005, pp. 791–797. [Online]. Available: http://www.comnets.rwth-aachen.de

[19] R. Chinnici, J.-J. Moreau, A. Ryman, and S. Weerawarana, "Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language," Published on the internet, May 2005, w3C Recommendation. [Online]. Available: http://www.w3.org/TR/2005/WD-wsdl20-20050510

[20] G. Gehlen, "Mobile web services - concepts, prototype, and traffic performance analysis," Ph.D. dissertation, RWTH Aachen University, Lehrstuhl fr Kommunikationsnetze, Aachen, Germany, Oct 2007. [Online]. Available: http://www.comnets.rwth-aachen.de

[21] M. Gudgin, M. Hadley, N. Mendelsohn, J.-J. Moreau, and H. F. Nielsen, "SOAP Version 1.2 Part 1: Messaging Framework," Published on the internet, Jun. 2003, w3C Recommendation. [Online]. Available: http://www.w3.org/TR/2003/REC-soap12-part1-20030624/

[22] G. Gehlen, F. Aijaz, and B. Walke, "An enhanced udp soap-binding for a mobile web service based middleware," in *Proceedings of IST Mobile Summit 06*. Myconos, Greece: ComNets, Faculty 6, RWTH Aachen University, Germany, Jun 2006, p. 8. [Online]. Available: http://www.comnets.rwth-aachen.de

[23] ——, "Mobile web service communication over udp," in *Proceedings of the 64th IEEE Vehicular Technology Conference*, IEEE. Montral, Canada: IEEE, Sep 2006, p. 1. [Online]. Available: http://www.comnets.rwth-aachen.de

[24] G. Gehlen and R. Bergs, "Performance of mobile Web Service Access using the Wireless Application Protocol (WAP)," in *Proceedings of World Wireless Congress 2004*. San Francisco, USA: University Aachen, Communication Networks, 05 2004, pp. 427–432. [Online]. Available: http://www.comnets.rwth-aachen.de/guge-pub.html

[25] SunLabs, *Sun Small Programmable Object Technology (Sun SPOT) Developers Guide*, v3 purple release ed., Sun Microsystems, Inc., 4150 Network Circle Santa Clara, CA 95045 U.S.A. 650 960-1300, October 2007. [Online]. Available: http://www.sunspotworld.com/docs/Purple/spot-developers-guide.pdf

[26] J. Postel, "User Datagram Protocol," RFC 768 (Standard), Aug. 1980. [Online]. Available: http://www.ietf.org/rfc/rfc768.txt

[27] M. Gudgin, M. Hadley, and T. Rogers, "Web Services Addressing 1.0 - Core," Published on the internet, May 2006, w3C Recommendation. [Online]. Available: http://www.w3.org/TR/ws-addr-core/

[28] F. Aijaz, H. Bilal, and W. Bernhard, "Towards peer-to-peer long lived mobile web services," in *Proceedings of the 4th International Conference on Innovations in Information Technology*. Dubai, UAE: IEEE, Nov 2007, p. 5. [Online]. Available: http://www.comnets.rwth-aachen.de

[29] J. Fuller, M. Krishnan, K. Swenson, and J. Ricker, "Oasis asynchronous service access protocol (asap)," 2005. [Online]. Available: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=asap