An innovative Hybrid Architecture and design for Wireless Sensor Networks:

Muhammad S. Aslam, Susan Rea, Dirk Pesch
NIMBUS Centre for Embedded Systems Research
Cork Institute of Technology
Ireland
{muhammad.aslam,susan.rea,dirk.pesch}@cit.ie

ABSTACT: *The issues in Wireless Sensor Networks (WSNs), are challenging as well as their applications demand for good services, distribution, modularity and intelligence. WSNs are characterized by varying software architectures, which has led to the development of a plethora of WSN middlewares, management and reconfiguration protocols. To ensure the potential of evolving WSN technologies and software architectures, it is required that the users can move away from traditional approaches towards an adaptable service oriented hybrid architecture, which should ideally run on the plate for wireless sensor devices, to minimize redundant protocol stack overhead. More importantly, the middleware should provide distributed machine to machine communication and implementation level model driven architectures should describe how the WSN works at each level of the network. In our work we present the compelling arguments for this vision of future WSNs and demonstrates how some of these concepts have been implemented on a significantly large physical WSN in NIMBUS Center of Embedded Systems Research Cork Ireland. Besides, this work provides an overview of implementation of the hybrid native architecture (HNA) on sensor devices and its collaborative functionality with a model based Open Framework Middleware (OFM) for WSN.*

Keywords: Wireless sensor networks, Network architecture, Open framework middleware, Hybrid native architecture

Received: 1 November 2010, Revised 29 November 2010, Accepted 5 December 2010

1. Introduction

Modern WSNs consist of number of wireless sensor devices sometimes called "*nodes*". A node can be a low-end or high-end device with different storage capacity, processing capabilities and software architecture. These nodes form networks to collect sensory data and transfer it to a base-station or sink sometime refer as gateway. The data can be used by number of different applications managing and monitoring the network. In certain environments these networks also comprise of actuators forming Wireless Sensor and Actuator Networks (WSANs).

Classification of WSNs from the software perspective is more complicated where there are operating systems, applications running on nodes and data aggregation and control applications running on sinks; however this classification is gradually disappearing by distributing the roles in the network therefore allowing middlewares to become part of the equation. Middlewares itself are complex to define where Bernstein [1] define middleware *"as a general purpose service that sits between platforms and applications. By platform, we mean a set of low level services and processing elements defined by the processor architecture and Operating Systems (OSs) API"*. According to [2] "*the scope of middleware for WSN is not restricted to the sensor network alone, but also covers devices and networks connected to the WSN*". [3] gives classification of middleware using dimensions where middleware such as Mate [4], VM*[5], DAVIM [6], SwissQM [7], Smart Messages [8], Agilla [9], SensorWare [10] are based on Virtual Machine and functionally span on limited matrices with a focus on nodes in sensor networks and technology. Hourglass [11], IrisNet [12], GSN [13], Cougar [14], SINA [15], DSWare [16], TinyDB [17], and Astrolabe [18] are some of the middlewares based on data-centric system which focus on multiple level of the middleware

classification matrix spanning from sensor network to backend and providing centralized management to the end system. A relatively modern middlewares such as Gridkit [19], AutoSec [20], MILAN [21], Adaptive- Middleware [22], MidFusion [23], TinyCubus [24], TinySOA [25], and RUNES [26] provide adaptability however their focus is clearly at the node level, therefore making nodes more adaptive to change.

From the functional analysis of these middleware we found that there were no comprehensive framework for the middleware which emphasize holistically on all aspect of the middleware classification matrix such that the middleware should be distribute as per role and data accumulation, fusion and communication should be service based and one of significant issue to address is the ability of the middleware to manage, configure, deploy and monitor the network. Furthermore the middleware framework should also focus on Machine-to- Machine (M2M) interaction allowing distribution of network where different application outside the network should be able to perform operation individually e.g. Monitoring, diagnostics, reasoning, network deployment etc.

In this paper we will give an overview of the OFM [27] a comprehensive middleware framework which allows use of modern paradigms of enterprise level system with examples of implemented case studies which combines existing middleware approaches to create an extensive WSN management system capable of interacting with other systems and the paper also proposes a HNA for nodes in conjunction with OFM as an alternative to the traditional layered protocol stack.

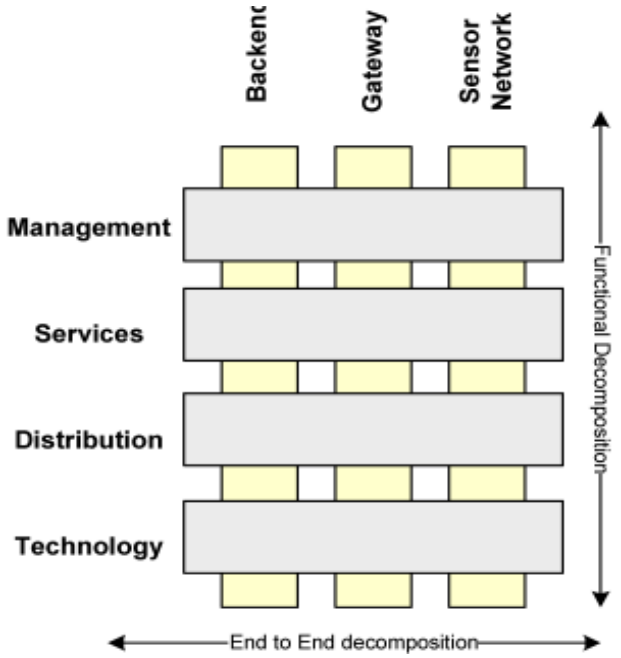

Figure 1. Middleware Classification Matrix [3]

2. Open Framework Middleware (OFM)

OFM is an experimental framework design of middleware for wireless sensor network, initially intend for network of wireless and wired sensors, however we observed that the implementation have scope of expansion which may include RFID, Bluetooth and other wireless devices. OFM mainly consist of three key elements i.e. Model, Core (Gateway middleware), Micro middleware.

Models define the network e.g. a model can define the topology of the network, sensing system for a node or localized reasoning in a node etc. The Core is the brain of OFM where it reads models, interpret them and use the information to activate services of the network at gateway and node level. The Core also provides M2M communication for distributed application accessing the network. Micro middleware is master program based on parameters which can be changed to change the functionality of the node; therefore a redeployment of the software on the node is not required. OFM is a use case in a number different on going projects in the domain WSN. Some of them are “Topology Adaption in Smart Building” [28]. Wi-Design/Wi-Manage and Network Monitoring and Reconfiguration Tool (NMRT).

2.1 Topology Adaption in Smart Building

One of the first use cases of the OFM where it was used with a system which analyze node faults in WSN such as low battery, route error and send a remediation by using reasoning based rule engine to OFM. The OFM then use this information to update the network routing by updating topology information on the nodes and at the OFM Core. The information send to the OFM is in form of models. Figure 2 shows an overview of the use case.

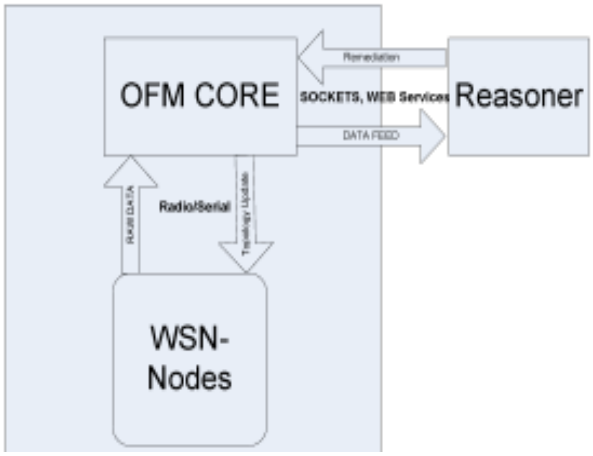

Figure 2. Integration of OFM with Reasoning System

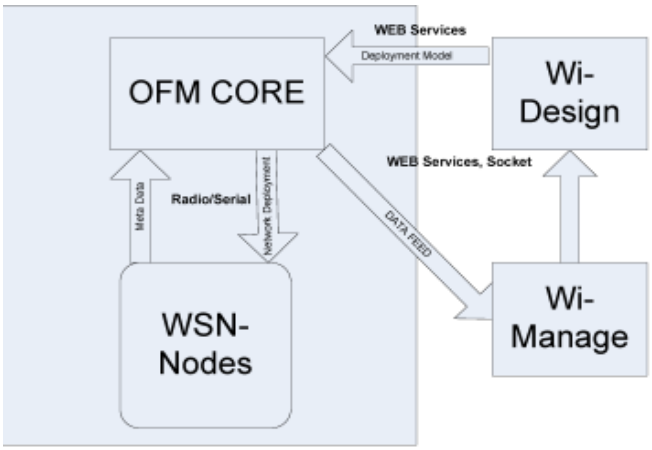

Figure 3. OFM Wi-Design and Wi- Manage

2.2 Wi-Design/ Wi- Manage

Ideally Wi-Design and Wi-Manage is WSN deployment, optimization and management tool which defines the optimized positioning of nodes in WSN, describes the topology by giving routing tables to each node and monitors parameters such as RSSI and link quality for keeping a check and balance on the integrity of the network and periodically update WSN by creating new deployment update models. Figure 3 shows the working process of Wi-Design and Wi-Manage with OFM.

2.3 NMRT

NMRT is another tool which uses OFM to integrate with the WSN. It is a simple network monitoring and reconfiguration tools which monitors sensory information such as light, temperature, battery level. This tool assumes that the WSN has already been deployed by third party tool such as Wi-Design. The tool also provide a method to reconfigure some of the parameters such as putting localized reasoning in node or activating or deactivating sensing and changing the sensing intervals e.g. “reporting battery level low when certain condition applies” or “deactivate temperature sensing” are some of the reconfiguration actions in NMRT. Figure 4 gives a simple overview of the NMRT working with OFM.

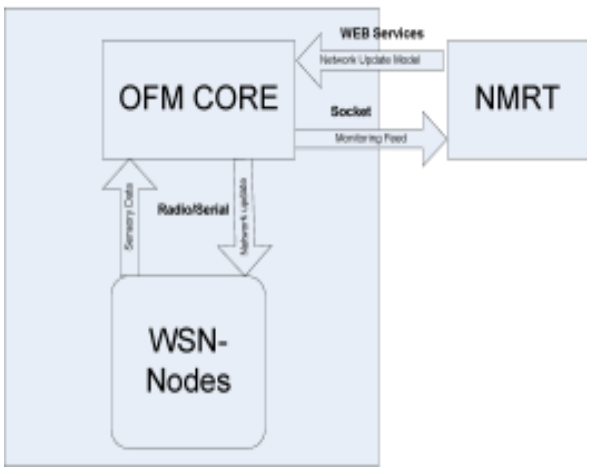

Figure 4. NMRT Tool with OFM

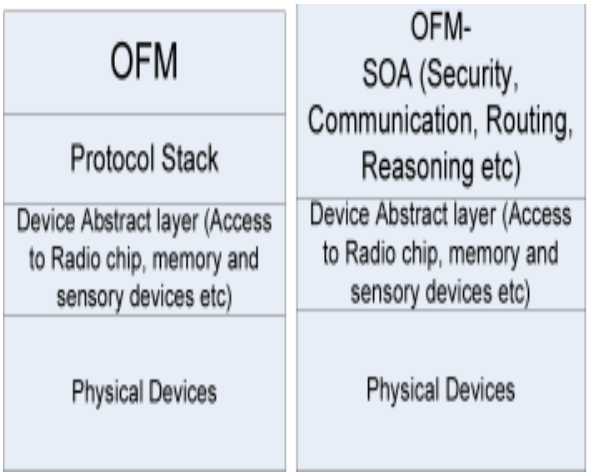

Figure 5. OFM Node vs OFM HNA node

3. OFM Hybrid Native Architecture

OFM is a distributed, service orientated model driven system; however the core issue facing OFM comes from the node level where nodes limit the function of what OFM is intended for i.e. using models to define WSN at all level of the network. In OFM, the Core and Models are high-level system which support standards such as the models are defined in eXtensible Markup Language (XML) [29] and most of the Core is based on java language running on open source java implementation servers, however nodes are very versatile where there are number of different operating systems which provides single and multi thread capabilities, Some operating systems provide modular update such as SOS [30]. OFM currently support only highly level nodes which support object oriented paradigm and that is one to reason SunSPOTs [31] were used in most of the use cases. Initially the problem of making the system adaptive was solved by creating a parametric master program which contains number of different services and each service can be activated by setting the parameter value send by models; however the system still relies on the underlying communication and operation stack as shown in figure 5. For OFM to be more efficient a lower level of access to hardware is required by removing the rigid layered traditional protocol. Protocol layers provide unnecessary overheads with its complex nested level abstraction and provide no easy way for modification and implementation. For example defining a radio packet with custom headers is one of the issues. In fact a protocol stack complying with a specific standard does not allow the upper layer to modify any information related to communication. In a protocol stack each layer is encapsulated in the layer below therefore putting unnecessary load on the performance due to linear initialization where information is passed from the top layer to bottom adding extra load to the information at each layer which may not be required at all.

Previous research [32] has highlighted the limitations associated with the traditional layered protocol stack approach and has shown that it restricts network software modularity. The rigidity of the layered protocol stack approach and the lack of information sharing between protocol layers impede optimal network performance as shared layer information is a prerequisite for network optimization in wireless heterogeneous environments.

For OFM to fulfill its claims, hybrid architecture is proposed which removes the stack based protocol layers and places emphasis on running a service oriented OFM micro middleware over the device abstraction level. Figure 5 shows the OFM HNA Node.

3.1 Services

One of the core principles of OFM is using ondemand services. In HNA the node consists of number of services which provides different functionality e.g. communication service handles the communication of data packets, security services check the integrity of these packets. Some of the main on-demand services are described as follows:

Communication Service: This service creates datapackets and uses custom headers. It also directly connects with the Physical layer to send data. The service also manages fragmentation of the datapackets to optimize delivery. In OFM HNA an intraservice communication mechanism is provided where a service can call other services.

Security Service: In an environment where datapackets require encryption this service can be activated. The encryption algorithm can be defined in models during network deployment or network update.

Reasoner Service: This service can be used to create localized rule for nodes. These rules can validate certain conditions and generate event against them. A simplified reasoning service is already a part of existing OFM micro middleware. The purpose of the reasoner service is to provide flexibility of the functional decomposition where condition monitoring, alerts, decision and intelligent system works at node level, therefore making the network more efficient. Localized reasoning system specifically comes handy in WSANs where the network cannot afford delays caused by reasoning applications running outside the network. The reasoning service also provides an opportunity for control applications to distribute their operations and delegate certain functionality to the node, however reasoning services have high computing requirements allowing it to work optimally on highend nodes such as SunSPOT platform.

Routing Service: This service is responsible for managing routing. The routing service may contain sets of pre-build routing systems such as AODV, LQR, however the goal is to make this service flexible enough to create a custom routing systems. For example in the NMRT project a point-to-point routing system was used which is based on real time static routing where a node's routing table contains the address of the next hop based on statically downloaded routing tables is defined at the network design phase, therefore gives some customization of the routing system. OFM HNA will provide more adaptability in implementing the custom routing by working at lower level of the node close to the physical system.

3.2 Master Program

The Master program is one of the key elements of OFM nodes. This is a program which is deployed only once on a node device. This is a parametric program where the functionality can be modified by changing the values of it parameters. The master program contain set of every possible instruction a node device can perform, therefore removing the overhead caused by redeployment of a program on node, which significantly reduces the cost of operation in the network, in terms of bandwidth use and energy usage. The Master program contains all the services available to the node where each service is activated on demand by the master program. For a program to implement every possible or most of the possible operation space can be an issue, however with the advent of high-end devices with reasonably large storage capacity it is no longer an problem but for low-end nodes with limited storage the functionality of the master program is reduced by removing certain services and compacting the operation. Figure 6 shows how OFM HNA master program works in conjunction with OFM where model are processed through the OFM core and then deployed on the node in the form a parametric map. The master program reads the parametric map and loads or unloads the requested services.

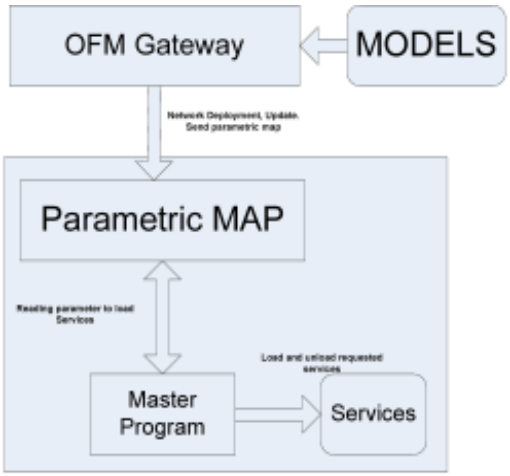

Figure 6. OFM HNA and OFM Interaction

3.3 Implementation.

The implementation of the OFM HNA is challenging where there are number of different type of nodes available running single image operating system such as TinyOS [33], modular operating system as SOS and without an operating system such as SunSPOTs. In most of the cases the line of abstraction between the operating system and the application is very difficult to understand in node. In order to implement the OFM HNA a standard abstraction system is required which should give access to the available node resources. OFM HNA does not require native access to the device; however a well abstracted resource access of the physical device is sufficient. For example the .Net MicroFramework [34] is a framework which can run "on the plate" allowing access to the node resource which it runs on. Similarly the Squawk [35] a java based virtual machine works in the same way. Such tools provide excellent opportunity for effective implementation of the OFM HNA.

4. Conclusion

This paper briefly analysis the core issue of WSN with a focus on middlewares. An overview of the implementation of a model based middleware solution is also provided in this paper. The approach in this paper focuses on the composable services which can easily be modified without redeployment. The paper also made reference to limitations of the traditional protocol stack approach and proposed a solution in the form of HNA which not only deals with a system consisting of distributed services interacting directly with low level resources but also manages all other operational aspect of nodes.

The proposed HNA working with OFM provides holistic control of the WSN with flexibility, reliability and adaptability at all levels of the network by using models to deploy, update and manage a network at device, gateway and enterprise level where HNA will give broader control over the resources of the nodes in WSN.

5. Future Work

Future work consist of extending the OFM to OFM HNA at node level and an implementation of the device abstraction

framework such as the .Net MicroFramework on heterogeneous network in a physical testbed with 120+ nodes located in the NIMBUS Centre for Embedded Systems Research. We are also working on improving the model translation process and the way in which information is transferred at gateway level and node level. The focus is to have a comprehensive WSN system which is distributed and service based system that allows M2M interaction for WSN using web services and socket communications.

There is huge scope of work in OFM and OFM HNA. At present we are investigating the interaction of the OFM based system with external tools and systems that provide feedback on the operational aspects of OFM allowing us to focus on the specific targets such as topology control from a communications perspective and reasoning for sensor network diagnostics.

References

[1] Bernstein, P. A. (1996). Middleware: A Model for Distributed System Services, Communications of the ACM, Feb 1996, 39 (2).
[2] Chatzigiannakis, I., Mylonas, G., Nikoletseas, S. 50 ways to build your application: A Survey of Middleware and Systems for. Wireless Sensor Networks.
[3] Horre, W., Mattys, N., Michels, S., Joosen, W., Verbaeten, P.A survey of Middleware for Wireless Sensor Networks.
[4] Levis, P., Culler, D. (2002). Mate : a tiny virtual machine for sensor networks, SIGOPS Oper. Syst. Rev. 36 (5) 85-95.
[5] Koshy, R., Pandey, J (2005). Vm*: synthesizing scalable runtime environments for sensor networks, *In*: SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems. New York, NY, USA: ACM, 2005, p. 243-254.
[6] Michiels, S., Horré, W., Joosen, W., Verbaeten, P. (2006). Proceedings of the international workshop on Middleware for sensor networks, Melbrone, Australia.
[7] Müller, R. , Alonso, G., Kossmann, D. (2007). SwissQM: Next Generation Data Processing in Sensor Networks, *In*: Proceedings of the 3rd Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 7-10th 2007.
[8] Kang, Porlin., Borcea, Cristian., Xu, Gang., Saxena, Akhilesh (2004). Ulrich Kremer and Liviu Iftode, Smart Messages: A Distributed Computing Platform for Networks of Embedded Systems, *The Computer Journal* 47(4) 475-494.
[9] Bhattacharya, S., Fok, C-L., Lu, C., Roman, G-C (2008). MLDS: A Flexible Location Directory Service for Tiered Sensor Networks, Computer Communications, Special Issue on Advanced Location-based Services, Computer Communications 1160 -1172.
[10] Boulis, A., Han, C-C., Shea, R., Srivastava, M. B. (2007). SensorWare: Programming sensor networks beyond code update and querying, *Pervasive and Mobile Computing*, 3 (4) August.
[11] Shneidman, J., Pietzuch, P. , Ledlie, J., Roussopoulos, M.M., Seltzer, M. Welsh Hourglass: An Infrastructure for Connecting Sensor Networks and Applications, Harvard Technical Report TR-21-04.
[12] Gibbons, P. B., Karp, B.Y., Ke, S., Nath, S., Seshan, (2003). IrisNet: An Architecture for a Worldwide Sensor Web, IEEE Pervasive Computing, 2 (4) 22-33, Oct-Dec.
[13] Aberer, K. M., Hauswirth, A., Salehi, (2007). Zeroprogramming Sensor Network Deployment, Next Generation Service Platforms for Future Mobile Systems (SPMS), Japan.
[14] Bonnet, P., Gehrke, J. E., Seshadri, P (2001). Towards Sensor Database Systems, In Proceedings of the Second International Conference on Mobile Data Management. Hong Kong, January.
[15] Srisathapornphat ,C., Jaikaeo, C., Shen, C-C (2000). Sensor Information Networking Architecture, *In:* Proceedings. International Workshops on Parallel Processing, p.23-30.
[16] Li, S., Son, S., Stankovic,J.(2003). Event detection services using data service middleware in distributed sensor networks. In Proceedings of the 2nd International Workshop on Information Processing, *In*: Sensor Networks, April 2003.
[17] Madden, S., Franklin, M. J., Hellerstein, J. M.,Hong,W. (2005). TinyDB: An Acqusitional Query Processing System for Sensor Networks, ACM TODS,
[18] Van Renesse, R., Birman, K P., Vogels,W. (2003). Astrolabe: A robust and scalable tech nology for distributed system monitoring, management, and data mining, ACM Transactions on Computer Systems.
[19] Grace, P., Coulson, G., Blair, G.B., Porter, D. Hughes, Dynamic reconfiguration in sensor middleware, *In*: Proceedings of the international workshop on Middleware for sensor networks.
[20] Han, Q., Venkatasubramanian, N. (2006). Information Collection Services for QoS-Aware Mobile Applications,*IEEE Transactions on Mobile Computing*, May.
[21] Heinzelman, W.B. Murphy, A.L., Carvalho, H.S., Perillo, M.A (2004). Middleware to support sensor network applications, *IEEE Network* 18.

[22] Huebscher, M.C., McCann, J.A. (2004). Adaptive Middleware For Context-Aware Applications In Smart-Homes, *In*: Proceeding of the 2nd International Workshop on Middleware for Pervasive and Ad-Hoc Computing, Middleware, Canada, October.
[23] Heinzelman, W.B., Murphy, A.L., Carvalho, H.S., Perillo, M. (2004). A Middleware to support sensor network applications, *IEEE Network* 18. 6—14.
[24] Marrón, P. J., Minder, D., Lachenmann, A., Rothermel. K (2005). TinyCubus: An Adaptive Cross-Layer Framework for Sensor Networks, *it - Information Technology*, 47(2) 87-97.
[25] Rezgui, A., Eltoweissy, M.(2007). Service-Oriented Sensor- Actuator Networks, *IEEE Communications Magazine*, 45 (12) Dec.
[26] Costa, P., Coulson, G., Mascolo,C. et al (2005). The RUNES Middleware: A Reconfigurable Component-based Approach to Networked Embedded Systems, *In*: The proceedings of (PIMRC05), IEEE, Berlin, Germany, September.
[27] Asalm, M S., O'Regan, E., Rea, S.. Pesch, D. (2009). Open framework middleware: an experimental middleware design concept for wireless sensor networks, *In*: International Conference on Autonomic Computing, Proceedings of the 6th international workshop on Managing ubiquitous communications and services ,Barcelona, Spain.
[28] Brennan, R., Tai,W., O'Sullivan, D Aslam, M S., Rea, S., Pesch, D. (2009). Open Framework Middleware for Intelligent WSN Topology Adaption in Smart Buildings, *In*: International Conference on Ultra Modern Telecommunications & Workshops, St. Petersburg.
[29] XML. (2010). Extensible Mark-up Language, http://www.w3.org/XML/, last accessed 18th March 2010
[30] https://projects.nesl.ucla.edu/public/sos- 2x/doc/index.html, last accessed 18 March 2010
[31] SunSPOT, Sun Microsystems, Project SunSPOT ,website http://www.sunspotworld.com/docs/, last accessed 18th Oct 2009
[32] Braden, R., Faber, T., Handley, M. (2002). From Protocol Stack to Protocol Heap- Role-Based Architecture, HotNets I, Preceton, NJ, USA, Oct.
[33] TinyOS, http://www.tinyos.net/, last accessed 18th July 2008
[34] http://msdn.microsoft.com/enus/ netframework/bb278106.aspx, last accessed, 17th March 2010
[35] https://squawk.dev.java.net/, last accessed, 17th March 2010