International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391

Comparative Analysis of Wireless Operating System

Rajdeep Singh Shaktawat

Department of Computer Science

Abstract: ***In today's era, Wireless Sensor Network (WSN) has become one of the most challenging area of research as its application range from small data aggregation to large fields (IoT). Wireless Sensor Network is used for diverse applications like agricultural monitoring, environmental monitoring, military, security, health-care any many more. Operating system (OS) plays a vital role in creating an efficient and reliable distributed application over the network. Over the years, many operating systems have emerged in the field of WSN which ease in the development of an application. This paper discusses present operating system for wireless sensor network and their major advantages with issues. It also addresses the architecture design of various WSN operating systems like Tiny OS, Contiki, and Lite OS. Paper concludes with the comparative analysis of all WSN operating system on various parameters.***

Keywords: Wireless Sensor Network (WSN), Operating System (OS), embedded operating system, System on chip (SoC), Real Time Operating System (RTOS)

1. Introduction

Advances in the area of Micro-electro-mechanical system (MEMS) and wireless communication technology leads to the miniaturization of low power sensor nodes. The nodes has the capability of sensing, performing various computations and ease of wireless communication. The collection of sensors will form an wireless sensor network. Each sensor node in a sensor network contains a micro controller chip, sensors, a radio transceiver and a memory as shown in Figure 1, a typical sensor node. Each node would sense its environment and would communicate with other nodes using multi-hop communication. The sensor node can monitor any type of environment and thus can be used for habitat monitoring, military, traffic control, home automation, agricultural monitoring and so on. The energy provided by the battery or any power source and there is main memory which is limited in size and used for data storage, these two are considered as most significant and the most vital resource of sensor node. The micro-controller used in a sensor node performs various tasks, processes data and controls the functionality of other components. Each sensor is a best example of a System on Chip (SoC). The deployment of sensors depends upon the application in which it is used.

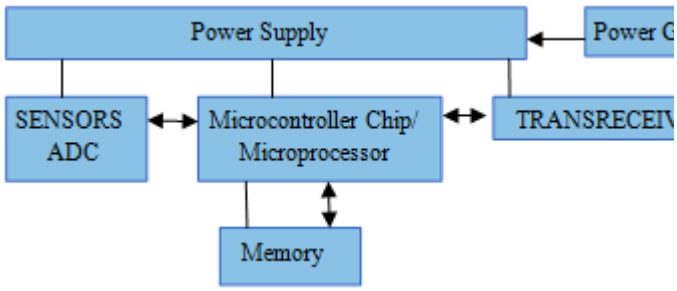

Figure 1: Architecture of Sensor Node

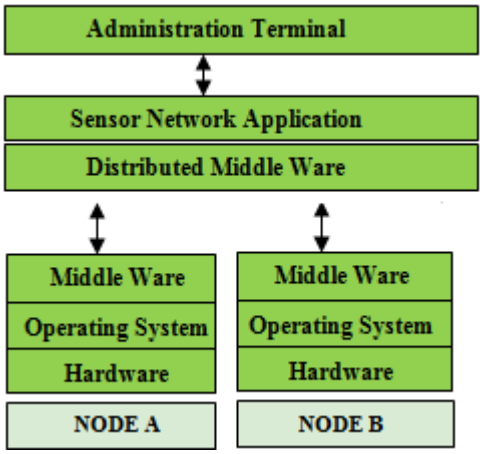

Figure 2: Software Architecture of Wireless Sensor Network

Software architecture of Wireless Sensor Network shown is Figure 2, depicts the overall software architecture of wireless sensor network. Each wireless sensor node in wireless sensor network has an operating system, which act as an middleware between the hardware of the sensor node and the application running over operating system. Many different applications might be deployed on various different nodes in a network, which work to provide specific services to the upper layer, the distributed middle ware work to provide co-ordination between these services running over different nodes within the same network. Nodes deployed at different location interact with this distributed middleware to accomplish the functions provided within the sensor network application. Basic functions of operating system include allocation and revoke of resources, make application interact with the hardware devices, raising the interrupt management, concurrency control, task scheduling and networking support. The admin terminal is considered to be isolated from this architecture it acts just as an interface to fetch result from sensor network application. The distributed middleware handles tasks for the entire network and it acts as network service coordinator [4][23][28][27][29][18].

The traditional operating system which is designed for workstation and personal computing needs ample of resources, It is the system software which works between the application software and hardware of devices. This could not be the case scenario with sensor nodes in WSNs. Traditional OS functions are therefore to manage processes, memory, CPU time, file system, and devices. There are embedded operating systems like VxWorks [39] and WinCE [40], but none of them is designed especially for data-centric WSNs with constrained resources. Sensors mainly have a slow processor and a limited small memory. There are many more parameters which should be kept in mind while designing the operating system for WSN nodes.

This paper investigates various challenging issues which arise while designing wireless sensor operating system. We have examined some existing operating systems for WSNs. The paper contributes by discussing various strengths and weakness of WSN operating system. The paper is further divided into sections, which discusses the design issues related to the operating system of WSN and then presents some existing operating systems which include TinyOS, Contiki, Nano-RK, MANTIS, LiteOS. At last comparative analysis of various WSN operating systems is presented on various parameters.

2. Issues in Designing an Operating System

Considering various special characteristics of sensor nodes, there is a requirement of different type of operating system for wireless sensor network. Following are various issues needed to be considered when designing operating system for WSN [14][7][6][37][5][1].

A. Process Management and Scheduling

A traditional operating system provides a separate memory space for each of its processes to execute. The process utilizes its own address space to execute its process. Various storage variables are used in this separate address space, data is manipulated in this space to fetch out the information according to the desired process. This method causes multiple data copying and context switching between processes. This method cannot be reliable in WSN as WSN network has a limited amount of memory and power. The operating system in sensor network should be optimize enough to provide efficient resource management mechanisms in order to allocate microprocessor time and memory space. This scheduling and allocation of processor time and limited memory should be subjected to some fair allocation scheme. [11][30]

B. Memory Management

In traditional operating system, memory is allocated exclusively to each process which helps in the execution of the task . In WSNs, as sensor nodes have small amount of memory, we need a different method, which can reduce memory requirements and should be capable of sharing data. [22][17]

C. Kernel Model

There are various event-driven models and finite state machine (FSM) models are used to design the micro kernels for the WSN. The event-driven model may serve the purpose of event-driven systems; an event may comprise receiving a packet, transmitting a packet, detection of an event of interest, alarms about energy depletion of a sensor node etc that would be beneficial for WSN. The FSM-based model in WSN is suitable to realize concurrency, reactivity, and synchronization. [25]

D. Energy Efficiency

In WSN, every node does not have constant power source, all are occupied with the limited power supply. With this limited power supply, the sensor node cannot work continuously for years. So the WSN operating system should be able to do optimize power management so that each sensor node remains active for its maximum life span, which helps to expand the system lifetime and improve its performance. For example, the operating system can put the node to sleep when to network is idle and wake up when any event occurs which can be interrupted by hardware. [33][36]

E. Application Program Interface

In WSN each sensor node is provided by API or application programming interface which connect the underlying hardware with the user. This may allow access and control of hardware directly, to be optimize system performance. WSN API provides a well-defined and easy-to-use way to collect data from WSN nodes, and give commands to them. Also, service discovery mechanisms and attribute based queries are enabled. [21]

F. Code Upgrading and Reprogramming

Code Reprogramming enables users to extend or correct functionality of a sensor network after deployment, at a low cost. The behavior of sensor nodes and the algorithms on which they are working may be needed to be adjusted or changed either for their functionality purpose or for energy conservation, the WSN operating system should be able to reprogram and upgrade. [3][26]

G. Limited Memory

The limited amount of memory which accounts to few kilobytes on a sensor node necessitates the OS to be designed with only required features. It is a fundamental characteristic of a sensor network operating system and the primary reason why so many sophisticated embedded OS cannot be easily ported to the sensor nodes. [17]

H. Real-Time Guarantee

Most WSN applications are tend for observation or examination in nature. They are time-sensitive. Thus, the packets which are to be transmitted are on timely basis. This approach requires real-time guarantee of data transmission. [12]

I. Reliability

In most of the applications, the on-site reliability of a sensor node is not guaranteed as sensor networks are deployed once and left unattended to operate for a long period of time. The reliability of the operating system is of great concern to facilitate developing complex WSN software, with ensuring the correct functioning of WSN systems. [34]

3. Existing Operating System for Wireless Sensor Network and Its Architecture

The basic function of the WSN is to collect information and to support certain application specific task of the deployment. For working, WSN needs different sensor node or mote. Currently, commercially available sensor nodes are categorized into four groups. [19][20][24]

1) Specialized sensing platforms i.e Spec
2) Generic Sensing platforms i.e Berkeley motes
3) High-bandwidth sensing platforms i.e iMote
4) Gateway platform i.e Stargate

Although these nodes have different characteristics, their basic hardware components are same like a physical sensor, a microprocessor, a memory, a radio transceiver and a battery. Therefore, these hardware components should be organized in a way that makes them work correctly and effectively without a conflict in support of the specific application for which they are designed. So, in that scenario, each node requires operating system that can control the hardware, provide hardware abstraction to the application software and the fill the gap between application software and hardware. Here, we discuss the architecture of different operating systems for Wireless Sensor Network like Tiny Os, Nano – RK, MANTIS, LiteOs and Contiki. [7] [37] [4] [10] [8][9]

A. Architecture of Tiny OS

Tiny OS [20] [38] was developed in 2000 by the UC Berkeley and is one of the earliest operating system for the Wireless Sensor Network. Tiny OS is component base, application specific operating system. [29] It follows Monolithic architecture class and uses an event – based model to support high level of concurrent application in a very small amount of memory. It also supports multithreading called as TOS Thread. Tiny OS footprint has fit in only 400 bytes of memory. It supports non-primitive FIFO (First-In-First-Out) algorithm for scheduling jobs. It provides static memory management. Tiny OS manages shared resources using any of the mechanisms like Virtualization and Completion Events. The main drawback of this OS is that it does not provide any support for real time application. Figure.3 shows the basic architecture of Tiny OS.

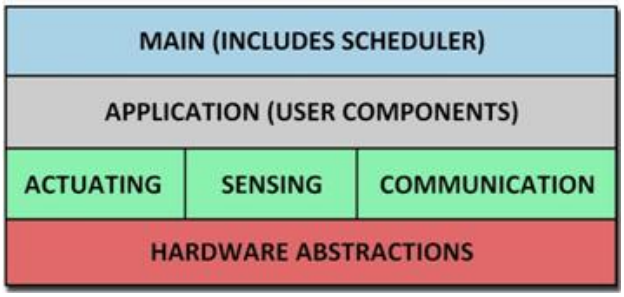

Figure 3: Architecture of Tiny OS

B. Architecture of Nano-RK

Nano- RK[13][16] is Real Time Operating System (RTOS) from Carnegie Mellon University with the purpose of running micro control of sensor network. It uses 2 KB RAM and 18 KB of ROM for performance. Its architecture is based on Monolithic kernel architecture model. To facilitate application developer it provides fully preemptive multi-threading support. Nano-RK provides priority scheduling at two levels: first at the process level and second at the network level. Nano-RK provides static memory management and light weight communication protocol stack similar to the socket. Using mutexes and semaphores serialized access, we can share resources. As Nano-RK is a Real Time Operating System (RTOS), it provides rich support for real time application. Figure.4 shows the basic architecture of Nano-RK.

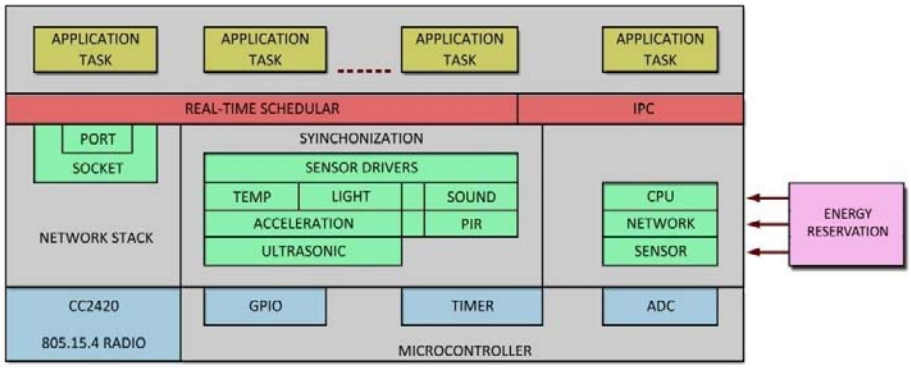

Figure 4: Architecture of Nano-RK

C. Architecture of MANTIS

MANTIS OS[29][35][15] is based on Layered architecture model. MultimodeAl system for NeTworks of In-situ wireless Sensor (MANTIS) provide multithreading for Wireless Sensor Network. MANTIS foot print fit in 500 bytes which include kernel, scheduler and network stack. MANTIS uses primitive based scheduling with multiple priority class, and it uses the round robin priority within the each class. It also allows dynamic memory management. Here, resource sharing is possible using semaphores. Figure.5 shows the basic architecture of MANTIS.

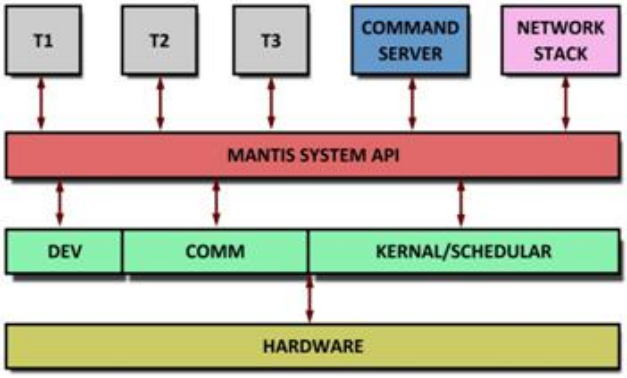

Figure 5: Architecture of MANTIS

D. Architecture of LiteOS

LiteOS[29][31][32] was developed by University of Illinois with the goal of providing UNIX like environment for programming in WSN application. Footprint of LiteOS is an 8 MHz CPU, 128 bytes of ROM and 4Kbytes of RAM. LiteOS follow Modular architecture model. Basically, LiteOS is partitioned into three subsystems: LiteShell, LiteFS and Kernel [29]. LiteShell is a UNIX-like shell that provides support for shell commands meant for file management, process management etc. A second subsystem is LitsFS which itself is file system. LiteFS mounts all neighboring sensor nodes as a file. And last subsystem is Kernel, which provide concurrency in the form of multithreading, support for dynamic loading, priority scheduling, etc. LiteOS is multitasking and multithreading OS. It follows priority based round robin policy for scheduling and supports dynamic memory management. LiteOS provides communication support in the form of files. Figure.6 shows the basic architecture of LiteOS.

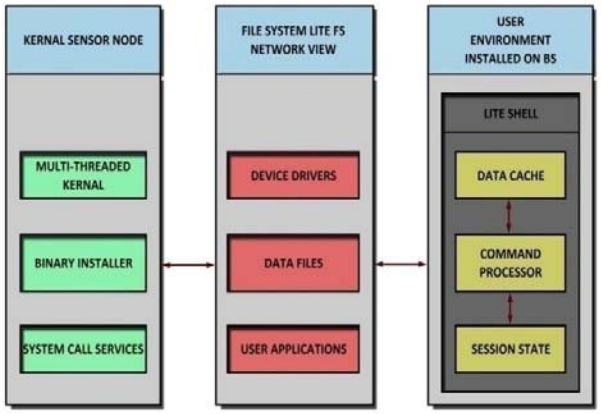

Figure 6: Architecture of LiteOS

D. Architecture of Contiki

Contiki is an open source operating system for network embedded system. [2] It follows modular architecture. It uses event driven with optional threading facility for processes. As Contiki is event driven operating system, it does follow any sophisticated scheduling algorithms. Scheduling is based on the event fired. It supports dynamic memory management and supports number of protocols for communication. Contiki provides serialized access for resources sharing. It does not give any support for real-time applications. Figure.7 shows the basic architecture of Contiki.

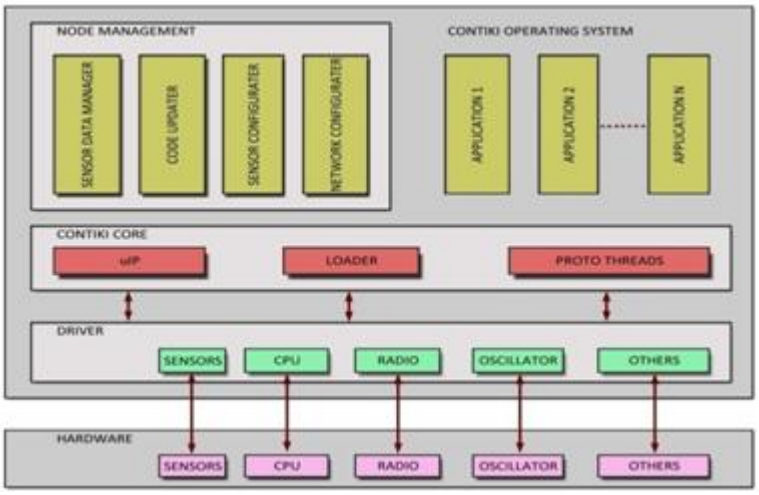

Figure7: Architecture of Contiki

4. Comparison Table for Different Operating System for WSN

In this section, we present a comparison of different OSes for WSN. We have done this comparison on the basis of various features discussed in each operating system in the above section.

Table 1: Comparison of different OSs for Wireless Sensor Network

Parameters/ WSN OS	Tiny OS	Nano – RK OS	MANTIS OS	LiteOS	Contiki OS
Architecture	Monolithic	Monolithic	Layered	Modular	Modular
Protocol Support	Active Message	Socket like abstraction for networking	At Kernel level COMM layer Networking layer is user level	File base communication	uIP, uIPv6 and Rime
Communication Security Support	Yes (Tiny Sec)	No	No	No	Yes (Contiki Sec)
File System	Single Level File System support	No	No	LiteFs	Coffee file System
Scheduling	FIFO	Rate Monotonic and rate harmonized scheduling	Five Priority Class	Priority based Round Robin Scheduling	Event are fired as they occur
Resource Sharing	Virtualization and Completion Events	Serialized Access through mutexes	Through Semaphores	Through Synchronization Primitives	Serialized Access
Memory Management	Static	Static	Dynamic	Dynamic	Dynamic
Memory Protection	Yes	No	No	Yes	No
Threading support	Yes	Yes	Yes	Yes	Yes
Event based Programming Support	Yes	No	No	Yes	Yes
Simulator	TOSSIM, Power Tossim	NA	AVRORA	AVRORA	Cooja, MSPSim, Netsim
Programming Language Support	Nes C	C	C	LiteC++	C
Sensing Platform support	Mica, Mica2, MicaZ, TelosB, Tmote, XYZ, IRIS, TinyNode, Eyes, Shimmer	MicaZ, FireFly	Mica2, MicaZ, Telos	Micaz, IRIS, AVR, MCU	Tmote, TelosB, ESB, AVR, MCU, MSP430 MCU,
Support for real time application	No	Yes	Up tp some extent at process scheduling level	No	No
Database support	Yes(Tiny DB)	No	No	No	No
Static/Dynamic System	Static	Static	Dynamic	Dynamic	Dynamic
Open Source	Yes	Yes	Yes	Yes	Yes
Publication year	2000			2008	2004
web site	http://www.tinyos.net	http://www.nano-rk.org	http://mantisos.org	http://www.liteos.net	http://sics.se/contiki

5. Conclusion

This paper discusses various wireless sensor network operating system, it also address various issues that has to be accounted while desiging the operating system for sensor network. Paper also represents various architecture design of some popular WSN operating system. A comparative analysis of various WSN operating system on the basis of various paramters is also included in paper which will help and inspire the researchers to design more robust and efficient operating system for WSN. The OS developers and the OS users will be able to know the features of various existing operating sytems of sensor network and can select the best suitable operating system for their application.

References

[1] A. K. Dwivedi, M. K. Tiwari, O. P. Vyas, "Operating Systems for Tiny Networked Sensors: A Survey", *International Journal of Recent Trends in Engineering, 2009,Vol. 1, No:2, pp. 152-157, ISSN 1797- 9617.*

[2] B.Gronvall, T.Voigt, "Contiki- a lightweight and flexible operating system for tiny networked sensors", EmNets, 2004.

[3] Adam Dunkels, Niclas Finne, Joakim Eriksson, Thiemo Voigt, "Run-Time Dynamic Linking for Reprogramming Wireless Sensor Networks", Sensys, 2006.

[4] Adi Mallikarjuna Raddy, D Janakiram , G Ashok Kumar, "Operating System for Wireless Sensor Networks : A Survey Technical Report", International Journal of Sensor Networks, 2009, Vol. 5, Issue 4, pp. 236-355, ISSN: 1748-1287.

[5] Ajay Jangra, Swati, Richa, Priyanka, "Wireless Sensor Network (WSN): Architectural Design issues and Challenges", (IJCSE) International Journal on Computer Science and Engineering, 2010, Vol. 2, No: 9, pp. 3089-3094, ISSN : 0975-3397.

[6] Anil Kumar Sharma, Surendra Kumar Patel, Gupteshwar Gupta, "Design Issues and Classification of WSNs Operating Systems", nternational Journal of Smart Sensors and Ad Hoc Networks (IJSSAN), 2012, Vol. 2, Issue: 3,4, pp. 71-75, ISSN No. 2248-9738.

[7] Antonio Augusto Frohlich, Lucas Franicsco Wanner, " Operating Systems Supports for Wireless Sensor Networks", Journal of Computer Science, 2008, pp. 272-281, ISSN 1549-3636.

[8] Chih-chieh Han, Ram Kumar, Roy Shea, Eddie Kohler, Mani Srivastava, "A Dyanamic Operating System for Sensor Nodes".

[9] Cintia B. Margi, Marcos A. Simplico Jr, Mats Naslund, "Impact of Operating Systems on Wireless Sensor Networks (Security) Applications and Testbeds", Proceedings of 19th International Conference on Computer Communications and Networks (ICCCN), 2010, pp. 1-6, ISSN : 1095-2055.

[10] D.Manjunath, A Review of Current Operating System for Wireless Sensor Networks.

[11] David Shuman, Mingyan Liu, "Optimal Sleep Scheduling for a Wireless Sensor Network Node", Fortieth Asilomar Conference on Signals, Systems and Computers, 2006. Pp. 1337 – 1341, ISSN: 1058-6393.

[12] Emanuele Toscano, Orazio Mirabella, Lucia Lo Bello, "An Energy-efficient Real-Time Communication Framework for Wireless Sensor Networks", Real Time Networks, 2007.

[13] Eswaran, A. Rowe, A.Rajkumar, "Nano-RK : An Energy - Aware Resource -Centric RTOS for Sensor Network", 26th IEEE Real-Time Systems Symposium, 2005.

[14] Frank Golatowski, Jan Blumenthal, Matthias Handy, Marc Haase, Hagen Burchardt, Dirk Timmermann, "Service-Oriented Software Architecture for Sensor Networks", In Proc. Int. Workshop on Mobile Computin, 2003.

[15] H.Abrach, S.Bhatti, J.Carlson, H.Dai, J.Rose, A.Sheth, B.Shucker, J.Deng, R.Han, "MANTIS: System Support for MultiModAl NeTworks of In-situ Sensors", International Workshop on Wireless Sensor Networks and Applications, 2003, ISSN: 1549-3474.

[16] Hojung cha, Sukwon Choi, Inuk Jung, Hyoseung Kim, Hyojeong Shing, Jaehyun Yoo, Chanmin Yoon, "RETOS: Resilient, Expandable and Threaded Operating System for Wireless Sensor Networks", The International Conference on Information Processing in Sensor Networks (IPSN 2007), pp. 148-157, ISBN: 978-1-59593-638-7.

[17] Hong Min, Sangho Yi, Yookun Cho, Jiman Hong, "An Efficient Dynamic Memory Allocator for Sensor Operating Systems", Proceedings of the 2007 ACM symposium on Applied computing, 2007, pp. 1159-1164, ISBN:1-59593-480-4.

[18] I.F. Akyildiz, Weilian Su; Sankarasubramaniam, Y.; Cayirci, E, "Wireless sensor networks: a survey", Communications Magazine, IEEE , 2002, Vol.40, pp.102-114, ISSN: 0163-6804.

[19] J.Hill, M.Hortion, R.Kling, L.Krishnamurthy, "The Platforms Enabling Wireless Sensor Networks", Communications of the ACM, 2004, Vol.47 No. 6, pp. 41-46, ISSN · 0001-0782.

[20] J.Hill, r.Szewezyk, A.Woo, S.Hallar, D.Culler, K.Pister, "System Architecture Directions for Networked Sensors", ACM SIGOPS Operating Systems Review, 2000, Vol. 34 No. 5, pp. 93-104, ISSN 0163-5980.

[21] Jari K. Juntunen, Mauri Kuorilehto, Mikko Kohvakka, Ville A. Kaseva, Marko Hännikäinen, Timo D. Hämäläinen, "WSN API: Application Programming Interface for Wireless Sensor Networks", The 17th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'06), Vol. 8, No. 6, ISSN : 1796-2056.

[22] John A. Stankovic, "Wireless Sensor Networks", Computing & Processing (Hardware/Software) Journal, 2008, vol. 41, issue 10, pp. 92-95, ISSN : 0018-9162.

[23] Kazem Sohraby, Daniel Minoli, Taieb Znati, "Wireless Sensor Networks: Technology, Protocols and Applications", Wiley Publications, 2007, ISBN 978-0-471-74300-2.

[24] Lalit Saraswat, Pankaj Singh Yadav, "A Comparative Analysis of Wireless Sensor Network Operating Systems", International Journal of Engineering and Technoscience, 2010, Vol. 1, No:1, pp. 41-47, ISSN: 1861-2121.

[25] Lin Gu, John A. Stankovic, "t-kernel: Providing Reliable OS Support to Wireless Sensor Networks", In Proc. of the 4th ACM Conf. on Embedded Networked Sensor Systems, 2006, pp.1-14.

[26] Milosh Stolikj, Pieter J. L. Cuijpers, Johan J. Lukkien, "Energy-aware reprogramming of sensor networks using incremental update and compression", The 3rd International Conference on Ambient Systems, Networks and Technologies (ANT-2012), 2012.

[27] Mohamed Watfa, Mohamed Moubarak, Ali Kashani, "Operating system designs in future wireless sensor networks", Journal of Networks, 2010, Vol. 10, No: 1, pp. 1201-1214.

[28] Mokhtar Aboelaze, Fadi Aloul, "Current and Future Trends in Sensor Networks: A Survey", Second IFIP International Conference on Wireless and Optical Communications Networks, 2005. WOCN 2005, pp.551-555, ISBN: 0-7803-9019-9.

[29] Muhammad Omer Farooq, Thomas Kunz, "Operating Systems for Wireless Sensor Network : A Survey", Sensors, 2011, Vol.11, pp. 5900-5930, ISSN 1424-8220.

[30] Ping Yi, Ting Zhu, Bo Jiang, Bing Wang, Towsley, D., "DEOS: Dynamic energy-oriented scheduling for sustainable wireless sensor networks", IEEE International Conference on Communications (ICC), 2012, pp. 3335 – 3339.

[31] Q.Cao, T.F.Adbelzaher, J.A.Stankovic, "The LiteOS Operating System: towards Unix -like abstraction for wireless sensor networks", ACM IEEE IPSN, 2008.

[32] Qing Cao, Tarek Abdelzaher, John Stankovic, Tian He, "LiteOS, A Unix-like Operating System and Programming Platform for Wireless Sensor Networks".

[33] RATHNA. R, SIVASUBRAMANIAN. A, "Improving Energy Efficiency In Wireless Sensor Networks Through Scheduling and Routing", International Journal Of Advanced Smart Sensor Network Systems (IJASSN), 2012, Vol. 2, No. 1, pp. 21-27, ISSN : 2231 – 4482.

[34] Robin Kim, Junho Song, Billie F. Spencer, Jr, "Reliability Analysis of Wireless Sensor Networks", The 6th International Workshop on Advanced Smart Materials and Smart Structures Technology, 2011, ISSN 1660-9336.

[35] S.Bhatti, J.Carlson, H.Dai, J.Deng, J.Rose, A.Sheth, B.Shucker, C.Gruenwald, A.Torgerson, R.Han, "MANTIS OS: An embedded multithreaded operating system for wireless micro sensor platforms", ACM/Kluwer Mobile Network Applicaton Journal, 2005, Vol.10, pp.563-579, ISSN: 1572-8153.
[36] Shashidhar Rao Gandham, Milind Dawande, Ravi Prakash, S. Venkatesan, Energy Efficient Schemes for Wireless Sensor Networks with Multiple Mobile Base Stations, Wireless Networks, Springer, 2011, Vol. 17, Issue 8, pp. 1809-1819.
[37] Thang Vu Chien, Hung Nguyen, Thanh Nguyen Huu, "A Comparative Study on Operating System for Wireless Sensor Networks", ICACSIS, 2011, pp.73.78, ISBN 978-979-1421-11-9.
[38] "tiny os", http://tinyos.net
[39] "VxWorks OS Development", www.windriver.com/products/vxworks/
[40] WinCE, http://www.microsoft.com/windowsembedded/enus/windows-embedded.aspx