Wireless Sensor Network, **2009, 1, 257-267**
doi:10.4236/wsn.2009.14032 Published Online November 2009 (http://www.scirp.org/journal/wsn).

Wireless Sensor Network Management and Functionality: An Overview

Dimitrios GEORGOULAS, Keith BLOW
Adptive Communication Networks Research Group,
EE, Aston University, Aston Triangle, B47ET, United Kingdom
Email: *dimitriosgeorgoulas@yahoo.com*
Received March 23, 2009; *revised May* 5, 2009; *accepted June* 12, 2009

Abstract

Sensor networks are dense wireless networks of small, low-cost sensors, which collect and disseminate environmental data. Wireless sensor networks facilitate monitoring and controlling of physical environments from remote locations with better accuracy. They have applications in a variety of fields such as environmental monitoring; military purposes and gathering sensing information in inhospitable locations. Sensor nodes have various energy and computational constraints because of their inexpensive nature and adhoc method of deployment. Considerable research has been focused at overcoming these deficiencies through more energy efficient routing, localization algorithms and system design. Our survey presents the fundamentals of wireless sensor network, thus providing the necessary background required for understanding the organization, functionality and limitations of those networks. The middleware solution is also investigated through a critical presentation and analysis of some of the most well established approaches.

Keywords: Wireless Sensor Networks, Organization, Functionality, Middleware

1. Introduction

Wireless sensor networks have been identified as one of most important technologies for the 21st century [1]. As technologies advance and hardware prices drop, wireless sensor networks will find more prosperous ground to spread in areas where traditional networks are inadequate. The foundational concept which applies in a vast number of networks can be identified through the simple notion: Sensing Capabilities plus CPU Power plus Radio Transmission equals a powerful framework for deploying thousands of potential applications.

However, this notion is underlined by some complex and detailed understanding of each separate network components capabilities and limitations as well as understanding in areas of modern network management and distributed systems theory.

The primary goal of wireless sensor networks is to make useful measurements for as long as possible. To do this it is essential to minimize energy use by reducing the amount of communication between nodes without sacrificing useful data transmission. Each node is designed in an interconnected web that will grow upon the deployment in mind. Wireless sensor networks are highly dynamic and susceptible to network failures, mainly because of the physically harsh environments that they are deployed in and connectivity interruptions [2].

To make the wireless sensor network dream a reality, an architecture must be developed that will monitor and control the node communication in order to optimize and maintain the performance of the network, ensure that the network operates properly and control/instruct a set of cluster nodes without human intervention [3].

In order to develop a system architecture with the above characteristics, we focus explicitly on the functions and the roles of wireless network management systems. Additionally, we present the middleware concept as a novel solution to the limitations that wireless sensor networks inhabit. A number of network systems are presented, critical reviewed and categorized.

2. Network Management Systems

Around 1980s computer networks began to grow and be interconnected in a large scale. This growth produced problems in maintaining and managing those networks, thus the need of network management was realized. Today, networks are far more dynamic and interconnected

than before, especially in the area of wireless sensor networks, thus a managing infrastructure is one of the most basic requirement for monitoring and controlling such networks [4].

A network management system can be defined as a system with the ability to monitor and control a network from a central location. Ideally there are four key functional areas that this system must support [5]:

1) Fault Management: This area provides the facilities that allow the discovery of any kind of faults that the managed devices of the network will produce, determining in parallel the possible causes of such errors. Thus, the fault management function should provide mechanisms for error detection, correction, log reports and diagnostics preferably without the user interference.

2) Configuration Management: Responsible for monitoring the entire network configuration information and also having access to all the managed devices in terms of reconfigure, operate and shut down if necessary.

3) Performance Management: Responsible for measuring the network performance through analysis of statistical data about the system so that it may be maintained at an acceptable level.

4) Security Management: This area provides all those facilities that will ensure that access to network resources can not be obtained without the proper authorization. In order to do so, it provides mechanisms for limiting the access to network resources and provides the end user with notifications of security breeches and attempts.

Those four functional areas of network management are far more challenging and vital for a network that will consist of tiny sensors which can be supplied to a specific environment running applications such as habitat monitoring, microclimate research, medical care and structural monitoring [6]. For every sensor network application, the network is presented as a distributed system consisting of many autonomous nodes that cooperate and coordinate their actions based on a predefined architecture. Every node is assigned with a specific role inside the network such as data acquisition and processing. Also, nodes can be used as data aggregation and caching points in order to reduce the communication overhead [7].

3. The System Organization of a Sensor Network System

The organization of sensor network systems is based upon the approach that they will adapt in order to monitor and control the state of the wireless sensor network. There are four predominant approaches [8]:

- Passive Monitoring: The system role is to collect data during the lifetime of the network. The data will identify the state of the network in different time intervals without any action taking place during the data gathering. An analysis of the data will take place in later stages.
- Fault Detection Monitoring: The system dedicates its resources to identifying faults and errors during the lifetime of the network. All the information is gathered and reported back to the operator whose responsibility is to correct those problems in later stages. No action is taken by the system towards the resolution of those problems in real time.
- Reactive Monitoring: The system has a double role to accomplish during the lifetime of the network. Firstly, as we identified in the previous approaches, the collection of data that will provide information about the states of the network, is the main role. This time though, the system will be eligible to identify and detect any events and act upon them in real time mainly by altering the parameters of the fixed asset under its control.
- Proactive Monitoring: The system collects and analyzes all the incoming data concerned with the state of the network. Then an analysis is taking place similar to the one of the reactive monitoring with the big difference that certain events, described by the collected information, are stored. The system is then able to maintain better available network performance by predicting future events based on past ones.

Wireless sensor systems can be categorized according to their architecture which can be centralized, hierarchical or distributed [9]. The centralized one identifies the role of the base station as the most important one in the whole architecture. The base station will collect the information from all the nodes and will monitor and control the entire network. Benefits to this architecture can be found in areas of processing power and decision making. A base station with unlimited power resources can perform complex analysis of data and process a variety of information, reducing the weight of this energy consuming task from the nodes of the network.

The distributed architecture focuses on the deployment of multiple manager stations across the network usually in a web based format. Thus, each substation can coordinate its actions and co-operate based on knowledge that it can acquire from a neighboring substations. In this approach, the communication cost is less than the centralized one and more energy efficient since all the workload will be distributed evenly across the network. However, due to the scalability and complexity of wireless sensor networks it is proven quite difficult to manage and quite expensive in terms of memory cost.

The hybrid between the centralized and distributed approach is that of the hierarchical one. In this architecture we have the existence of substations in the network but this time no communication is allowed between them. The design tends to be cluster based, with the heads of the cluster be responsible for a set of network nodes in

terms of processing and transmitting information. All the cluster heads will report back to a single base station.

4. The Functionality of a Sensor Network System

The main functionality of sensor network systems is based on the theory behind network management systems, thus is focusing on two attributes those of monitor and control. In this section, we classify some well known sensor systems in terms of the functionality they provide inside the network. Figure 1 is demonstrating this classification.

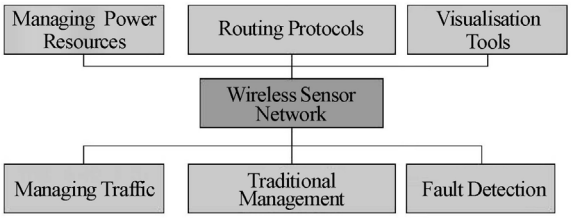

Figure 1. The functionalities of wireless sensor networks in different categories.

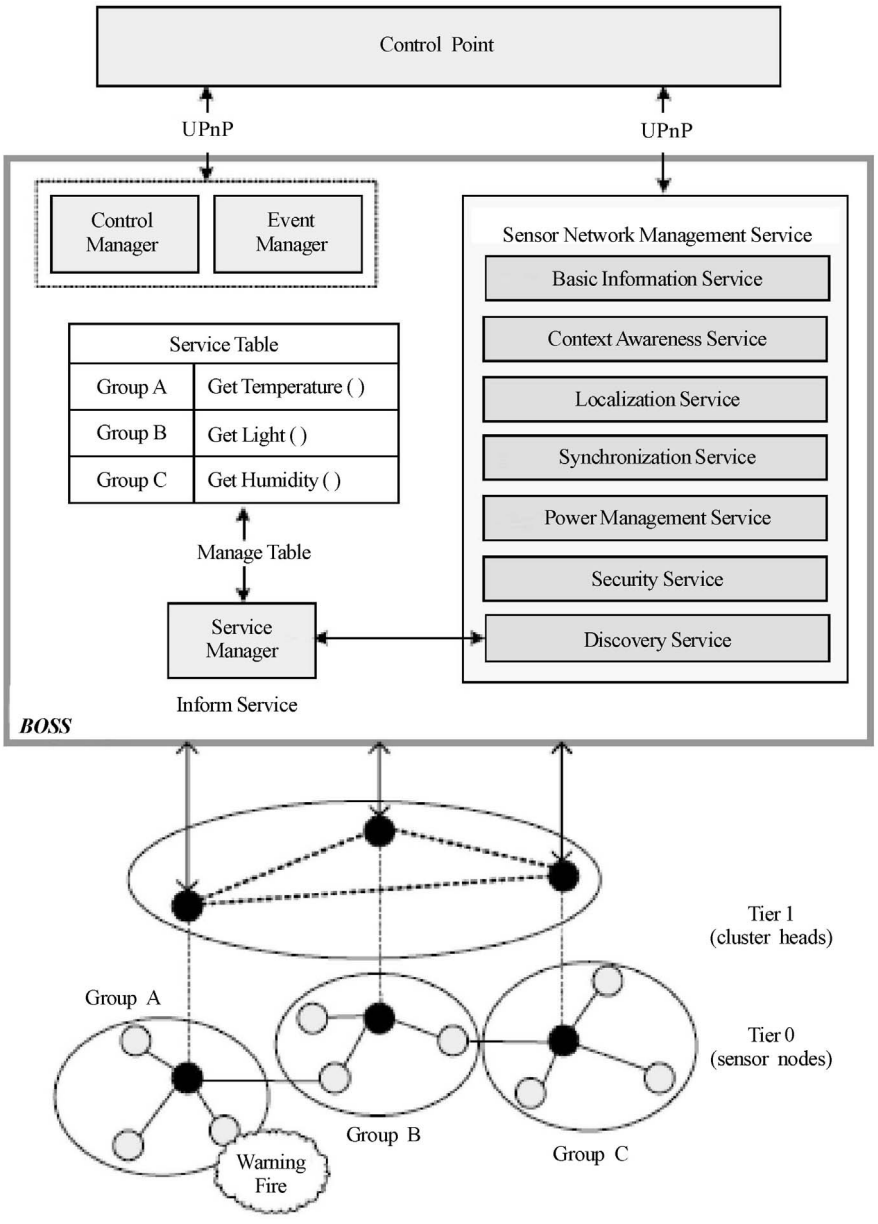

Figure 2. The BOSS architecture, song *et al* 2005.

Two characteristic examples of wireless sensor networks that are based on traditional management systems are those of MANNA [10] and BOSS [11]. MANNA provides a general architecture for managing a wireless sensor network by using a multidimensional plane for the functional, physical and the informational architecture of the network. The functional plane is responsible for the configuration of the application specific entities, the information plane is object oriented and specifies all the syntax and semantics that will be exchanged between the entities of the network and lastly the physical plane establishes, according to the available protocols profiles, the communication interfaces for the management entities that will be present inside the network.

The BOSS architecture, Figure 2, is based on the traditional method of the standard service discovery protocol, UpnP. With the UpnP protocol the user does not need to self configure the network and devices in the network automatically will be discovered. However, due to computational power consumption required by the devices and memory space allocation limitations, the protocol itself is not suitable for tiny sensor devices. BOSS architecture is overcoming this problem by acting as a mediator between UpnP networks and sensor nodes. In order to do that, it combines four different components: service manager, control manager, service table and a sensor network management service, under the same framework.

Routing protocols is another alternative way of monitoring and controlling a wireless network when they get embedded in an application with examples such as LEACH [12] and GAF [13].

LEACH is a routing protocol for users that want remotely to monitor an environment. The protocol is build upon two foundational assumptions. The first one acknowledges that the base station is at a fixed point and in a far distance from the network nodes and the second one assumes that all nodes in the network are homogeneous and energy constrained. In order to maximize the system lifetime and coverage, LEACH is using a set of methods such as distributed cluster formation with randomized selection of cluster heads and local processing. LEACH dynamic clustering method, splits time in fixed intervals with equal length. Also, it does not allow clusters and cluster heads to be at a fixed point inside the network. LEACH, dictates that once other sensors of the network receive a message they will join a cluster with the stronger signal cluster head.

GAF, which stands for geographic adaptive fidelity, focuses its architecture on the extension of the lifetime of the network by exploiting node redundancy. This node redundancy is achieved by switching off unnecessary sensor nodes in the network without any effect on the level of routing fidelity.

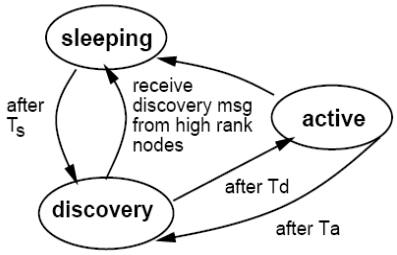

Figure 3. The GAF nodes state transitions, xu *et al* 2001.

GAF recognizes three transition states for the nodes, Figure 3, active, sleeping and discovery. Initially all nodes in the network are in a discovery state. This means that all nodes will turn their radio on and exchange discovery messages in order to identify neighbor nodes in the same grid. When a node is active it will set a timeout, Ta, that will determine for how long it will stay in that state before it returns back to the discovery state. While active, the node periodically re-broadcasts its discovery message at time intervals, Td. The sleeping state is regulated by a time interval Ts which is dependent upon the application. GAF assumes that sensor nodes can identify their location in the forming virtual grid with the use of GPS cards.

Systems, such as WinMS [14] and Sympathy [15] focus more on the importance of fault detection in a wireless sensor network. WinMS uses a novel management function, called systematic resource transfer, in order to provide automatic self-configuration and self-stabilization both locally and globally for the given wireless sensor network. This function allows the network, in case of a failure, to have a predetermined period of time where nodes will listen to their environment activities and self-configure. No prior knowledge of the topology of the network is necessary. WinMS, uses a TDMA-based MAC protocol, called FlexiMAC [14], in order to support resource transfer among nodes in the network. FlexiMac protocol provides synchronized communication between the nodes. Thus, it can adaptively adjust the network by providing local and central recovery mechanisms.

Sympathy is a tool for detecting and debugging failures in wireless sensor networks, but unlike WinMS it does not provide any automatic network reconfiguration incase of a failure. One of the main functionalities of the system is the collection and analysis of network information metrics such as nodes next hop and neighbors. By doing so, it is able to identify which of the nodes deliver insufficient data to the sink node or to the base station and locate the cause by reporting back to the end user. One of the major advantages of Sympathy is that it takes into account interactions upon multiple nodes however, by doing so it will require nodes to exchange neighborhood lists, something that has proven highly costly in terms of energy levels.

Another very useful functionality of wireless sensor systems is that of the visualization of the actual network. This ability of an end user to demonstrate graphical representations of the different states of the network at various time intervals can be found in systems such as the TinyDB [16] and MOTE-VIEW [17].

TinyDB is a distributed query processor for sensor networks. It uses an SQL like interface for collecting data from nodes in the given environment and also provides aggregation, filtering and routing of the acquired results back to the end user. With the use of a declarative language for specific user queries, TinyDB proves to be flexible in two domains. Firstly, all the queries that are generated are easy to read and understand and secondly the underlying system will be responsible for the generation and the modifications of the query without the query itself to need any modifications. In the core of the system we find a metadata catalog that identifies the commands and attributes that are available for querying.

The MOTE-VIEW system is an interface system between the end user and the deployed network of wireless sensors. Through this interface the user can make alterations to node characteristics in terms of radio frequency, sampling frequency and transmission power. The system architecture is based on four layers; data access abstraction, node abstraction, conversion abstraction and visualization abstraction layer. The data abstraction layer acts as the database interface where all the data is been stored. The node abstraction layer collects and stores all the nodes metadata which will create relational links with the database. All the raw data that is going to be collected from the nodes will be translated into understandable engineering units at the conversion abstraction layer. Finally, the visualization abstraction layer will provide to the end user displays of the data in forms of spreadsheets and charts.

MOTE-VIEW is a passive monitor system in that it does not provide any interpretation of the displayed graphical data on behalf of the user. However, in terms of network and other failures MOTE-VIEW does not provide any self-configuration scheme.

The resource management is one of the key aspects of every wireless sensor network. Systems such as the Agent Based Power Management [18] and SenOS [19] have been created with that concern in mind. The Agent Based Power Management is a system that builds its architecture upon mobile agents. These intelligent entities are set responsible for local power management processing by applying energy saving strategies to the nodes of the network. This agent-based scheme is suitable for applications where the state of the network is partial visible at a known time or location. One of the major concerns of this approach is that by minimizing the transmission power, the communication range of the nodes will be reduced accordingly, threatening the network connectivity.

SenOS is managing network power resources by instructing nodes to sleep when they are not active inside the network. To achieve this, SenOS adapts a dynamic power management algorithm known as DPM. The DPM algorithm, by observing events inside the network, can generate a policy for state transitions. Based on that, all redundant nodes are placed inside a cluster with only one node awake for a period of time per cluster while the others are in a sleep mode for conserving energy. The SenOS architecture is based on a finite state machine which consists of three components. Firstly, we have the kernel which provides a state sequencer and an event queue. The second component is a transition table and the final component is a call back library.

Siphon [20] and DSN RM [21] are two representative systems that provide traffic management functions in their architecture. Siphon, is based on a Stargate implementation of virtual sinks in order to prevent congestion at near base stations inside the network. These virtual sinks act as intermediates between the actual nodes and the base stations and they are distributed randomly inside the network. If at any point the rate of generate data increases beyond a level that exists in a predetermined threshold inside the system then the virtual sinks will redirect the traffic to other visible nodes. The visibility of the available nodes by the virtual sinks is one of the disadvantages of this approach, as there is a high probability that some nodes will be not covered by any virtual sink.

DSN RM (Distributed sensor network resource management) uses single radio nodes to evaluate each of their incoming and outgoing data rate and apply delay schemes to those nodes when necessary in order to reduce the amount of the traffic in the network. In every DSN there are a number of decision stations whose role is to act as data managers in a hierarchical format. However, the effectiveness of this technique is tightly bound on finding reliable data for every decision station inside the wireless sensor network that from its nature can provide inaccurate data during its lifetime due to connectivity and radio problems.

Table 1 presents a tabular evaluation of the currently available systems in terms of their organization and functionality.

5. The Role of Middleware in Wireless Sensor Networks

Many researchers have identified, that a middleware inside a wireless sensor network can establish a framework for bridging the gap between applications and low levels constructs such as the physical layer of the sensor nodes.

Table 1. Wireless sensor network systems evaluation based on designed criteria.

Wireless sensor Network Systems	Reactivity	Architecture	Function	Energy efficiency	Adaptability	Memory efficiency	Scalability
BOSS	Proactive	Centralised	Management System	Yes	Yes	Yes	Yes
MANNA	Proactive	Hierarchical	Management System	N/A	N/A	N/A	N/A
LEACH	Proactive	Distributed	Routing Protocol	Yes	Yes	Yes	Yes
GAF	Proactive	Distributed	Routing Protocol	Yes	Yes	Yes	Yes
WinMS	Proactive	Hierarchical	Fault Detection	Yes	Yes	Yes	Yes
Sympathy	Proactive	Centralised	Fault Detection	Yes	Yes	Yes	No
TinyDB	Passive	Centralised	Visualization Tool	Yes	No	Yes	Yes
MOTE- VIEW	Passive	Centralised	Visualization Tool	Yes	No	Yes	Yes
SensOS	Reactive	Hierarchical	Management resources	Yes	No	Yes	No
A. B. P. M	Proactive	Distributed	Management resources	Yes	Yes	Yes	No
DSNRM	Proactive	Hierarchical	Traffic Control	Yes	Yes	No	No
Siphon	Proactive	Distributed	Traffic Control	No	Yes	Yes	No

A middleware can be visualized as a network managing software mechanism that will create communication bonds with the network hardware, the operating system and the actual application. A fully implemented middleware should provide to the end user a flexible interface through which actions of coordination and support will take place for multiple applications preferably in real time.

This section identifies some of the most well known middleware approaches that have been developed in recent years and classifies them according to their programming paradigm. This classification presents middlewares as virtual machines based on modular programming, virtual database systems and adaptive message oriented systems.

The use of a virtual machine inside a middleware is a flexible approach since it can allow a programmer to partition a large application into smaller modules. The middleware will inject and distribute those modules inside the wireless sensor network with the use of a predefined protocol that will aim to reduce the overall energy and resource consumption. The main role of the virtual machine is to interpret those distributed modules. The communication protocol can be designed based on modular programming. The use of mobile code can facilitate an energy efficient framework for the injection and the transmission of the application modules inside the network.

The Mate [22] middleware is among those that use a virtual machine in order to send applications inside the wireless sensor network. The developers having identified the predominant limitations of wireless sensor networks such as energy consumption and limited bandwidth propose a new programming paradigm that is based on a tiny centric virtual machine that will allow complex programs to be very short. In order to achieve that, Mate's virtual machine acts as an abstraction layer with content specific routing.

This section identifies some of the most well known middleware approaches that have been developed in recent years and classifies them according to their programming paradigm. This classification presents middlewares as virtual machines based on modular programming, virtual database systems and adaptive message oriented systems.

The use of a virtual machine inside a middleware is a flexible approach since it can allow a programmer to partition a large application into smaller modules. The middleware will inject and distribute those modules inside the wireless sensor network with the use of a predefined protocol that will aim to reduce the overall energy and resource consumption. The main role of the virtual machine is to interpret those distributed modules. The communication protocol can be designed based on modular programming. The use of mobile code can facilitate an energy efficient framework for the injection and the transmission of the application modules inside the network.

The Mate [22] middleware is among those that use a virtual machine in order to send applications inside the wireless sensor network. The developers having identified the predominant limitations of wireless sensor networks such as energy consumption and limited bandwidth propose a new programming paradigm that is based on a tiny centric virtual machine that will allow complex programs to be very short. In order to achieve that, Mate's virtual machine acts as an abstraction layer with content specific routing.

Figure 4 presents Mate architecture and execution model. This high level architecture will enable the programming code to break up into small capsules of 24 instructions each that can self-replicate inside the network.

This architecture enables Mate to begin execution in response to a specific event such as a packet transmission or a time out. This is applicable through the three execution contexts that refer to an equal amount of events: clock timers, message receptions and message send re-

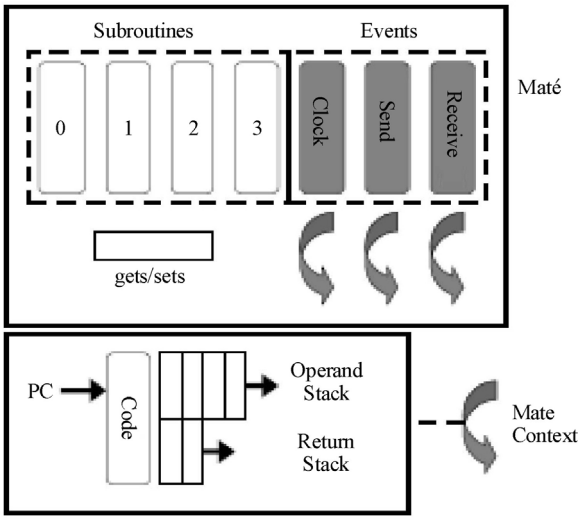

Figure 4. Mate architectural concept, P. Levis and D. Culler (2002).

quests. Each of the three contexts has an operant stack and a return address one. The operant stack will be used for instructions handling all the data while the return address stack will handle all the subroutine calls.

Every capsule that is sent inside the network includes a type and a version number. Based on that information, Mate can achieve easy version updates by adding a new number to the capsule every time a new version of the program is uploaded. However, Mate middleware suffers from the overhead that every new message introduces and also all the messages are transmitted by flooding the network in order to minimize asynchronous events notifications, raising issues with the energy consumption of every node inside the network.

Agilla [23] is a middleware that provides a mobile code paradigm for programming and making effective use of a wireless sensor network. Agilla applications consist of mobile agents that can clone or migrate across the network. The framework is based on the fact that each agent is acting as an autonomous entity inside the network allowing the developer to run parallel processes at the same time. Agilla is based on the Mate architecture in terms of the virtual machine specifications but unlike Mate, which as we described above divides an application into capsules flooding the network, Agilla uses mobile agents in order to deploy an application.

Figure 5 presents the Agilla model identifying the communication principle between two neighbor network nodes.

In every network node we can have one or more agents residing and working independently. Their intercommunication and coordination is established by local tuple spaces that are accessible by all the agents resident in that node and a neighbor list. The local tuple space is a shared memory architecture that is addressed by field-matching. A tuple can be defined as a sequence of data objects that is inserted into the tuple space of each node by every agent. These data objects will remain in the node regardless of the agent status. In due time, an agent can retrieve an old tuple by template matching. In order to do so the sending agent must generate a query for that tuple, matching the exact same sequence of fields. The neighbor list contains the addresses of neighboring nodes and is accessible for every agent in the network that wants to clone or migrate in a different location.

Based on these attributes, Agilla allows network reprogramming thereby eliminating the power consumption cost of flooding the network. However, the lack of a hierarchical communication model for the agent society and the absence of precise real location information of every node in the network can lead to deadlocks and memory management problems.

What is known as a database middleware will visualize the whole network as a virtual database system. The middleware in this case will provide the user with an interface for sending queries to the sensor nodes of the system to extract the desired data.

The Cougar middleware adopts the above approach by considering the extracted sensor data to be a virtual relational database. The developers by using an SQL-like language assume that the whole network is the database. The contents of such a database are stored data and the sensor data. The stored data is represented as a virtual relationship between the sensors that participate in the network and the physical characteristics. The sensor data which is the outcome of processing functions is represented as time series that will be adapted towards the query formulation.

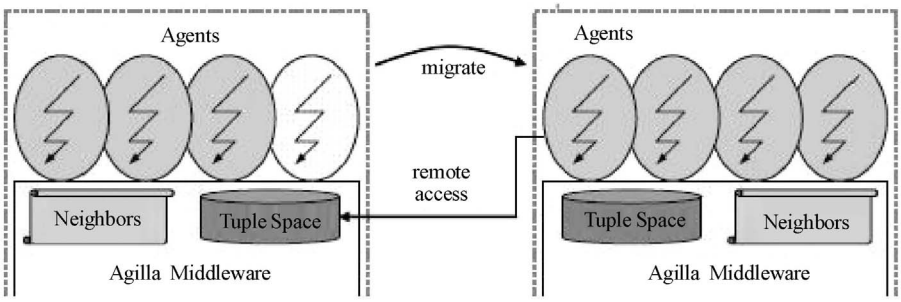

Figure 5. The Agilla architectural concept, C. Fok and G. Roman (2005).

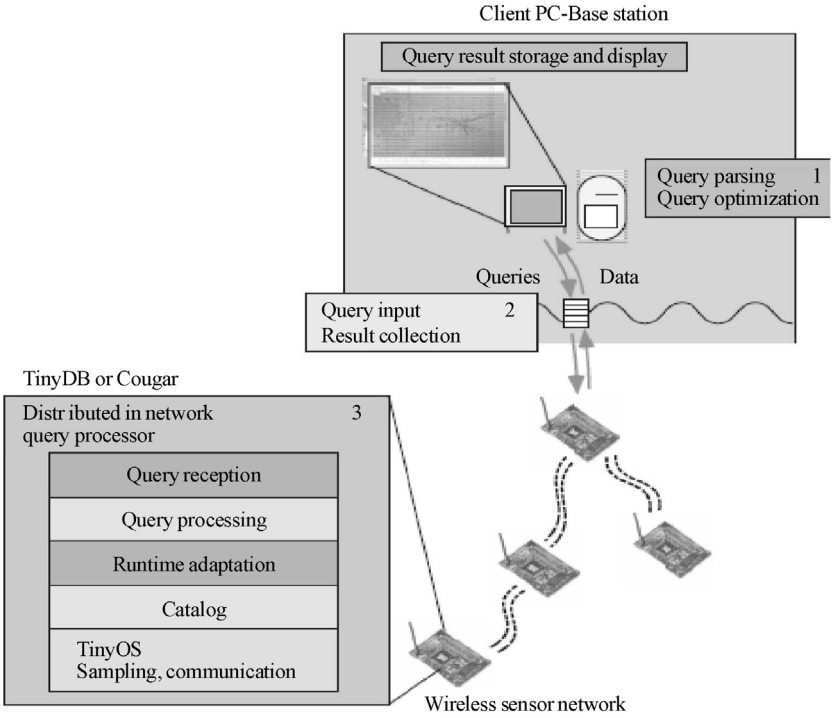

Figure 6. Cougar architecture for query processing, G. Gehrke and S. Madden (2002).

Figure 6 shows a block diagram that can explain the Cougar architecture for querying processing. One block presents the user end with the base station in the active role of transmitting and receiving queries from the wireless sensor network. The second block presents the distributed network query processor that consists of a number of abstract data types with virtual relationships with the operating system of the network. In an SQL query format of SELECT-FROM-WHERE-GROUP-INCLUDES Cougar can allow the user to access this object relational database which mirrors the actual network.

The third class of middleware is message oriented. Their core architecture is based on creating a communication model that will facilitate message exchanges between nodes and the sink nodes of the wireless sensor network. To achieve that most middlewares will adapt a publish-subscribe mechanism, an asynchronous communication paradigm which allows a loose coupling between the sender and the receiver saving precious power resources.

Mires [24] middleware provides such an asynchronous communication model. This model defines three distinct faces for the nodes resident in the network. Initially the network nodes will advertise their sensed data (topic). Using a multi-hop routing algorithm Mires will route those advertisement messages to the sink node. Lastly, a user application interface will select the desired advertised topics to be monitored.

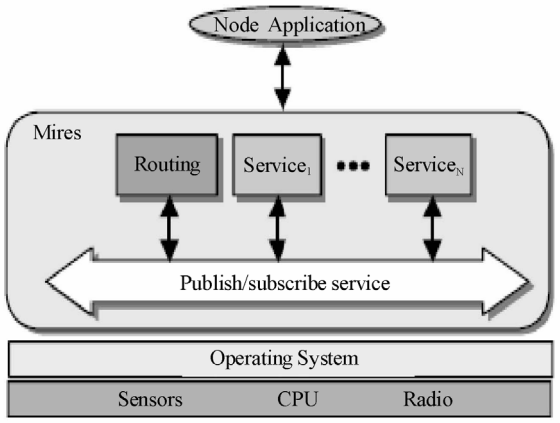

Figure 7. The Mires architecture, E. Souto and G. Vasconcelos (2004).

Figure 7 demonstrates the key characteristics of the Mires architecture. The bottom block consists of the hardware components of the node which are directly interfaced and controlled by the operating system. The middleware is placed on top of the operating system to implement its publish-subscribe communication model. This model is able to advertise the sensor data (topics) provided by the running application while it maintains a topic list provided by the node application. Mires send only messages referring to subscribed topics thus reducing like that the numbers of the transmitted packets and therefore saving energy.

6. Towards the Design of the In-Motes Middleware

In order to design and develop a successful middleware solution for a wireless sensor network that will be able to satisfy some if not all the functionalities of a network management system for monitor and control there are certain design criteria, which we described in the previous sections, and must be considered and brought forward in our design. In the following paragraphs we are going to present those design principles for a substantial middleware development.

A wireless sensor network consists of tiny devices that are battery powered and provided with a small CPU processor. Usually, and as we have already mentioned, they can be deployed in hundreds and typically in physical harsh environments. It is obvious, that after the deployment a physical contact for replacement or maintenance is highly unlikely. A middleware should be able to provide remote access to these nodes making sure that they will exhaust all their resources in terms of battery power and memory in a timely manner. Hence, one of the basic design principles for our middleware is the ability to manage limited power and resources.

Our approach [25] in order to satisfy the above design criterion is based in the creation of a flexible communication protocol between the nodes of our network and the base station. Thus, inspired from the GAF protocol that was described in the previous section, we are aiming to develop a protocol having node redundancy in mind, thus to regulate in an energy efficient manner which of the nodes of our wireless sensor network will be active and which ones will be in a sleep mode. Subject to our trials and the development of our middleware this protocol will be introduced both hand written in the middleware engine as well as it will be introduced as part of the running application.

Field trials such as the CodeBlue Project [26] and the Wireless Sensing Vineyards [27] identified that a wireless sensor network topology is subject to frequent changes due to factors such as device failure, interference, mobility and moving obstacles. Also, they proved that it is very possible that an application will grow in time, therefore mechanisms for a dynamic network topology should be available from the middleware. A middleware should be able to adapt to parameter changes caused by unexpected external factors of the environment and also provide mechanisms for fault tolerance and self configuration of the nodes inside the wireless sensor network [28].

Based on the above observations, and inspired from approaches similar to WinMS and Sympathy our middleware will incorporate some mechanisms for fault detection and prevention together with a flexible way of reconfiguring the network in real time [29]. As Sympathy is a fully automated system, we will be aiming to provide some kind of automatic mechanisms in our middleware for the above design criteria without though this to be our first priority.

Unlike traditional networks, sensor networks and their applications are real-time phenomena with dynamic resources. Upon deployment of an application, core parameters such as energy usage, bandwidth and processing power cannot be predefined due to the dynamic character of these networks. A middleware should be designed with mechanisms that should allow the network to run efficiently and as long as possible. Such mechanisms include resource discovery and location awareness for the nodes in the system. Low-level programming models must be introduced as well in order to bridge the gap between the running application and the hardware.

As we mentioned before, mechanisms that are predominant in traditional networks are not sufficient to maintain the quality of service of a wireless sensor network because of constraints such as the dynamic topology and the power limitations. A middleware should be able to provide and maintain the quality of service over a long period of time while in parallel to be able to adapt in changes based on the application and on the performance metrics of the network like these of energy consumption and data delivery delay.

Inspired both form the work of the Mate and Agilla middlewares we will introduce a mechanism that will allow mobile code to be transmitted inside our network and will be able to allow changes in network parameters such as the bandwidth of each node as well as application parameters such as switching between available sensors of each node. Thus, an architecture will be developed adapting technologies such as Linda-like tuple spaces and agents/mobile code transmission with the support of a virtual machine engine [30].

Wireless sensor networks can be widely deployed in areas such as healthcare, rescue and military, all of these are areas where information has a certain value and is very sensitive. The environments of those areas tend to be very active and harsh, increasing the chances for malicious intrusions and attacks such as denial of service. Traditional approaches and mechanisms used to secure the network cannot be applied in this kind of network since they are heavy in terms of energy consumption. A middleware must be able to provide a secure framework for deploying applications inside the network. During the life-cycle of the network the middleware should establish protective mechanisms to ensure security requirements such as authenticity, integrity and confidentiality.

Table 2 presents a tabular evaluation of the currently available middleware systems in terms of their organization and functionality.

Table 2. Wireless sensor network middleware systems evaluation based on designed criteria.

Project	Main features	Openness	Scalability	Mobility	Heterogeneity	Power Awareness	Easy of use
Mate	Mobile active Capsules, TinyQS, Byte code interpreter	Full	Full	Full	Partial	Full	Little
Agilla	Generic Agents, TinyQS	Full	Partial	Full	Partial	Little	Average
Cougar	Virtual Database, SQL like language	Partial	Partial	Partial	Partial	Partial	Full
Mires	nesC, TinyQS, message oriented	Full	Full	Partial	Full	Full	Full
In-Motes	nesC, TinyQS, agents, behavioral rules	Full	Full	Full	Full	Full	Full

7. Conclusions

This survey presents and demonstrates the wireless sensor networks as one of the predominant technologies for the 21st century. The foundational concept behind this technology is explained together with the limitations and barriers that are incorporated and need to be addressed in the process of making a wireless sensor network applicable for a number of useful applications in modern societies. The survey opens by presenting the foundational functions of a network management system. Functions that are based upon two main attributes those of monitor and control and are critical for every wireless sensor system. A number of wireless sensor network systems are critically reviewed providing an analytical explanation of their architecture and identifying pros and cons thus helping us draw some important lessons for our proposed system. The survey continues by presenting the middleware solution as a key player in overcoming the wireless sensor networks limitations and as our main methodology behind our proposed approach. A classification and analysis of the current middleware approaches for the wireless sensor networks is produced. The survey concludes by presenting the role and the key design principles our middleware approach.

8. References

[1] D. Culler, D. Estrin, and M. Srivastava, "Overview of sensor networks," IEEE Computer, pp. 41–49, 2004.

[2] J. Hill, R. Szewczyk, A. Woo, S. Hollar, and D. Culler, "System architecture directions for networked sensors," Architectural Support for Programming Languages and Operating Systems, pp. 93–104, 2000.

[3] A. Boulis, C. C. Han, and M. B. Srivastava, "Design and implementation of a framework for efficient and programmable sensor networks," International Conference On Mobile Systems, Applications And Services, pp. 187–200, 2003.

[4] Cisco Systems, "Network management systems organization," 2000, http://www.ddirv.lv/doc_upl/sazonovs1.pdf.

[5] W. Stallings, "Wireless communications and networks," ISBN-10: 0131918354, pp. 45–50, 2004.

[6] D. Culler, D. Estrin, and M. Srivastava, "Overview of sensor networks," IEEE Computer, pp. 41–49, 2004.

[7] I. F. Akyildiz, Y. Sankarasubramaniam, W. Su, and E. Cayirci, "Wireless sensor networks: a survey," Computer Networks, pp. 393–422, 2002.

[8] W. L. Lee, A. Datta, and R. Cardell-Oliver, "WinMS: Wireless sensor network-management system," CSSE Technical Report, UWA-CSSE-06-001, 2006.

[9] I. F. Akyildiz, Y. Sankarasubramaniam, W. Su, and E. Cayirci, "Wireless sensor networks: a survey," Computer Networks, pp. 393–422, 2002.

[10] L. B. Ruiz and J. M. Nogueira, "MANNA: Management architecture for wireless sensor networks," Communications Magazine, Vol. 41, No. 2, pp. 116–125, 2003.

[11] H. Song, D. Kim, K. Lee, and J. Sung, "UPnP-based sensor network management architecture," Proceedings of the second International Conference on Mobile Computing and Ubiquitous Networking, pp. 85–92, 2005.

[12] W. Heinzelman, A. Chandrakasan, and H. Balakrishan, "Energy efficient communication protocol for wireless microsensors networks," Proceedings of the Hawaii International Conference on System Science, pp. 10–17, 2000.

[13] Y. Xu, "Geography-informed energy conservation for ad hoc routing," Mobicom'01, pp. 203–212, 2001.

[14] W. L. Lee, A. Datta, and R. Cardell-Oliver, "WinMS: Wireless sensor network-management system," CSSE Technical Report, UWA- CSSE-06-001, 2006.

[15] N. Ramanathan, K. Chang, R. Kapur, L. Girod, and E. Kohler, "Sympathy for the sensor network debugger," Centre for Embedded Network Sensing, pp. 98, 2005.

[16] S. Madden, J. Hellerstein, and W. Hong, "TinyDB: In-network query processing in TinyOS," ACM Transactions on Database Systems, pp. 122–173, 2003.

[17] Touron, "Crossbow: moteview interface," Crossbow, 2005, http://www.xbow.com/Technology/UserInterface. aspx.

[18] R. Tynan, D. Marsh, D. O'Kane, and G. M. P. O'Hare, "Intelligent agents for wireless sensor networks," AAMAS 2005: 1179–1180.

[19] T. H. Kim and S. Hong, "Sensor network management

protocol for state-driven execution environment," Proceedings of the International Conference on Ubiquitous Computing, pp. 197–199, 2003.

[20] J. Wan, S. Eisenman, A. Campbell, and J. Crowcroft, "Siphon: overload traffic management using multi-radio virtual sinks in sensor networks," Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems, pp. 116–129, 2005.

[21] Zhang, "Network management in wireless sensor networks," 2001. http://www.csse.uwa.edu.au/~winnie/Network_Management _in_WSNs_.pdf.

[22] P. Levis and D. Culler, "Mate: A tiny virtual machine for sensor networks," Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems, San Jose, CA, USA, pp. 100–111, 2002.

[23] C. L. Fok and G. C. Roman, "Mobile agent middleware for sensor networks: An application case study," Proceedings of the 4th International Conference on Information Processing in Sensor Networks, pp. 382–387, 2005.

[24] S. Eduardo, G. Germano, and V. Glauco, "A message-oriented middleware for sensor networks," Proceedings of the 2nd workshop on Middleware for pervasive and ad-hoc computing, pp. 127–134, 2004.

[25] D. Georgoulas and K. Blow, "In-Motes: An intelligent agent based middleware for wireless sensor networks," Best Student Paper Award, Proceedings of the 5th WSEAS International Conference SEPADS 06, pp. 225–231, 2006.

[26] Welsh *et al*, "Wireless medical sensor networks in emergency response: Implementation and pilot results," IEEE International Conference on Technologies for Homeland Security.

[27] Holler, "Wireless sensing vineyards," 2008, http://camalie.com/WirelessSensing/WirelessSensors.htm.

[28] D. Georgoulas and K. Blow, "Making motes intelligent: An agent based approach to wireless sensor networks," In WSEAS on Communications Journal, pp. 515–522, 2006.

[29] D. Georgoulas and K. Blow, "In-Motes bins: A real time application for environmental monitoring in wireless sensor networks," Proceedings of the 9th IEEE/IFIP International Conference on Mobile and Wireless Communications Networks, pp. 21–26, 2007.

[30] D. Georgoulas and K. Blow, "Intelligent mobile agent middleware for wireless sensor networks: A real time application case study," AICT 2008, in Proceedings of 4th Advanced International Conference on Telecommunications: Programmable networks, active networks and mobile agents, protocol & standards, Athens, pp. 35–43, 2008.

