WSNMP: A Network Management Protocol for Wireless Sensor Networks*

Muhammad Mahbub Alam, Md. Mamun-Or-Rashid and Choong Seon Hong
Department of Computer Engineering, Kyung Hee University
1 Seocheon, Giheung, Yongin, Gyeonggi 449-701 Korea
{mahbub, mamun}@networking.khu.ac.kr, cshong@khu.ac.kr

***Abstract* — Wireless sensor networks are uniquely characterized by their limited resources and are deployed in remote and hostile environments. These highly dynamic networks are very prone to failure and are usually kept unattended. Therefore, proper management of WSN and its limited resources is highly desirable for an effective and efficient functioning of the network. In this paper, we propose WSNMP, a low overhead, hierarchical wireless sensor network management protocol. The proposed mechanism provides the methods to monitor the network states by collecting management data and accordingly control and maintain the network resources.**

***Keywords* — Network Management, Wireless Sensor Networks, Fault Management, Cluster Formation.**

1. Introduction

Network management is the task of detecting and resolving network problems both locally or remotely. It includes a set of functions and a network manager and agents. The manager acquires the state of the network from agents using various functions and accomplishes the task of configuration management, fault management, performance management, security management etc. Wireless sensor networks (WSNs) differ from traditional data networks in many ways. It has the unique characteristics of dynamic topology, infrastructure less, innumerable amount of traffic flow from simple periodic traffic to burst of message triggered by external events, energy and memory constrained. As a result, management techniques followed in traditional networks are impractical for WSNs.

Wireless sensor network is a collection of power and memory constraint small tiny devices and likely to operate under a very dynamic critical environment with applications such as environment monitoring, public safety, medical, transport and military. In disaster management type applications sensors are deployed in the inaccessible area and left unattended. As a consequence, network maintenance for reconfiguration, recovery from failure or solving technical problems by human intervention is impractical. While, sensor nodes may fail due to power constraint or physical damage or environmental interference. Therefore, sensor networks are in need of some management protocol to accomplish the task of integrated configuration, operation, administration and maintenance of all elements and services. We focus on applications that provide management schemes in terms of monitoring and controlling WSNs.

The management protocol should be capable of collecting information about node energy level, communication power, topology of the network, link state and coverage and exposure bounds of WSNs. Based on these information, management protocol can perform periodic sleep and awake (for power management), wireless bandwidth usage (traffic management), error recovery and network reconfiguration.

While designing management protocol architecture, following design criteria should be taken into account to adopt with the characteristics of sensor network: i) Low computation and message overhead ii) Minimal memory operations iii) Scalability iv) Application independence v) Robustness and fault tolerance vi) Energy efficiency.

In this paper we propose a management architecture protocol for wireless sensor networks that monitors the networks with minimum overhead, collect the management data and finally manages the network in an efficient way by periodically reconfiguring the network periodically. It also detects the fault of the network by identifying the non-response nodes and if necessary reconfigures the routing path.

The rest of the paper is organized as follows: Section 2 briefly describes about related works, Section 3 articulated network model and architecture. Section 4 elaborates the proposed mechanism. Section 5 discusses experimental evaluation of our proposed mechanism and finally we conclude in section 6.

2. Related Works

Based on operational behavior, management architecture of WSNs is classified into three categories [1]: centralized, distributed and hierarchical. Centralized and distributed systems maintain their traditional definition and being controlled by a central and distributed manager respectively. While hierarchical architecture performs the management tasks in a hybrid fashion. Centralized system executes complex management tasks in a central manager with unlimited resources. At a glance centralized system looks quite good for sensor network as the manager does not put any pressure to the energy constraint nodes deployed in the sensor field. However, it incurs a high message overhead (bandwidth and energy) for data polling, and this limits its scalability. Also, the central server is a single point of data traffic concentration and potential failure. Finally, if a network is partitioned, then nodes that are unable to reach the central server are left without any management functionality. Example of centralized system includes BOSS [3], MOTE-VIEW [5], and Sympathy [4].

* "This research was supported by the MIC under the ITRC support program supervised by the IITA" (IITA-2006-(C1090-0602-0002)).
* Dr. Hong is corresponding author.

Distributed architecture manages the task of management using multiple manager stations. Where, each manager controls a sub-network and communicates directly with other manager stations. Although distributed architecture has advantage over centralized architecture, but it is computationally too expensive for energy and memory constraint sensor network. Employing complex function execution in the deployed nodes may drain energy quickly and introduce network partitions. DSN RM [8], Node-energy level management [7], App-Sleep [6], and sensor management optimization [9] are the example of distributed management architecture which all require significant memory resources.

A variant of distributed architecture is mobile agent-based framework. Examples of this kind system include Mobile Agent-Based Policy Management [11], Agent-Based Power Management [10], and MANNA [2]. However, main disadvantage of this approach is it requires manual configuration and thus human intervention. This is impractical for sensor networks deployed in the inaccessible terrain.

Hierarchical management architectures also use multiple managers like distributed systems. There is a hierarchy among the managers in case of hierarchical management architecture. Each manager passes information from its management area to its higher-level manager, and also disseminates management functions received from the higher-level manager. Unlike other systems, DSN RM [8] and WinMS [12] allow individual nodes to act as agents and perform management functions autonomously based on their neighborhood states. Typical examples of this kind of architecture are AppSleep [6], TopDisc [13], STREAM [14], and SenOS [15].

3. Network Model and Assumptions

We consider a network of N sensing nodes, deployed with uniform random distribution over an area A. We consider a single sink in the network, placed at anywhere within the terrain. All sensors are static; we do not consider mobile sensors that form a dynamic ad-hoc network.
The network is homogeneous. All nodes have the same processing power and equal sensing and transmission range. Data generation rate of each sensing node is also assumed to be equal. Our proposed protocol and schemes are independent of the underlying network layer protocol. CSMA/CA is used as MAC protocol.

The network can be either event driven where nodes within the event radius generate traffic or periodic where sensors periodically generate data at a fixed rate. The sensed data eventually reach to the sink, forwarded by intermediary cluster heads and gateway nodes where a gateway is sensor node that connects two nearby cluster heads.

Our management model implements a central manger (CM) at the sink and an intermediate manager (IM) in each cluster head, while the agents are implemented at each sensor node. Central Manager at the sink collects management level information from the agents through the intermediate managers stationed at the cluster head.

4. Wireless Sensor Network Management Protocol (WSNMP)

4.1 WSNMP Architecture

The architecture of WSNMP is shown in Fig. 1, which represents the relationship among management services and management functionalities. WSNMP is hierarchical network management system where there tier architecture is used: the central manager is at the highest level and placed at the sink node, the intermediate manager works at the cluster heads and management agents are the normal sensor nodes. The intermediate managers are used to distribute management functions and collect and collaborates management data. However, they do not communicate with each other and work independently. They execute management functions based on their local network states whereas central network manager has the global knowledge of the network states and gather the global knowledge from the underlying network layers and intermediate network managers.

4.2 Network Configuration Management

The network configuration management service collects

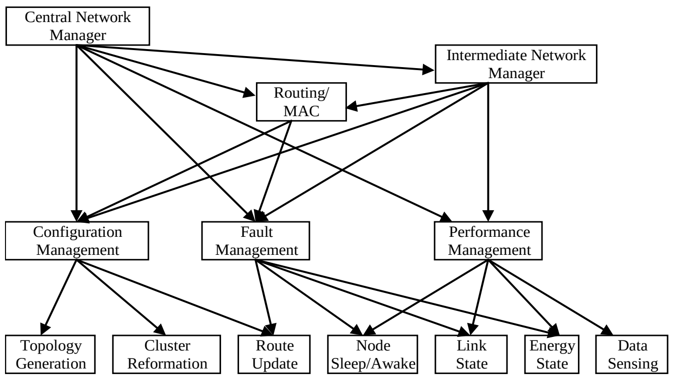

Figure 1. WSNMP architecture and relationship among different

information about the network states and based on that information it reconfigure the networks. In hierarchical sensor network one of the major issues is to create the cluster. Most of the existing proposals use flooding to create the clusters. However to keep energy use of the sensors at an equilibrium state, cluster heads and gateway nodes are to changed periodically, thus requires a re-clustering mechanism. If cluster reconfiguration is done by flooding following the initial process, then this creates not only huge energy consumption but also during the re-clustering period, usual data transfer may need to be stopped. In this section we describe an algorithm to generate the topology of the networks. Once the topology of the network is modeled, the CM can reconfigure the network with minimum overhead.

4.2.1 Cluster Formation

Many proposals exist in the literature for cluster creation in multi-hop wireless networks. Any of these clustering algorithms can be used for the initial cluster formation. The cluster formation algorithm proposed in [13], TopDisc is used to select the initial cluster heads and the gateway nodes.

TopDisc allows a minimal set of nodes to be active in maintaining network connectivity. The algorithm selects a set of distinguished nodes and forms a network topology based on the nodes' neighborhood information. Nodes listen to other nodes within their communication range to collect local neighbor information.

4.2.2 Topology Generation

Our topology generation algorithm works in two phases. In the first phase the cluster head (and so the IM) generates the topology within the cluster and sends the accumulated data to the sink. This first phase is based on the broadcast nature of the wireless networks. Just after the cluster formation, every member of the cluster informs the cluster head as a member of the cluster and at the same time sends their neighbor list. When any node hears the transmission of other it includes that node in its neighbor list. Once the cluster head hears the joining message from all of its neighbors it can build the topology within the cluster.

In Fig. 2(a) CH_1 is the cluster head and n_1 to n_5 are the members of the cluster. If n_1 sends the joining message first then it neighbor list is empty. But n_5 and n_2 add n1 in their

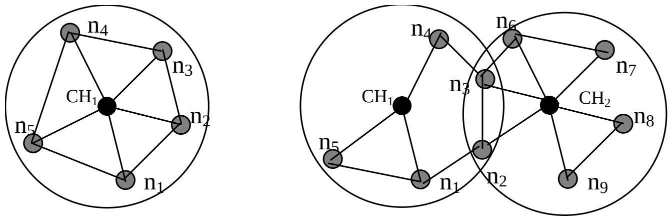

Figure 2. (a) Topology within a single cluster (b) Toplogy for neighboring cluster.

neighbor list by hearing the transmission of n_1. So when n_2 sends the joining message its neighbor list includes n_1, thus CH_1 knows that n1 and n_2 are neighbor. When CH_1 receives the joining message from all the members, it can complete the topology within the cluster. However the complete topology generation may not be possible just by receiving the joining message, because if two nodes of two neighboring cluster sends the joining message simultaneously, then neighboring nodes in the overlapping region cannot extract their transmission and will not include them in their neighbor list. Therefore, the generated topology will be incomplete. However during the usual data transmission time they can build the complete neighbor list. So when a new node is added in the neighbor list of one node, it piggybacks it in the next data packet and eventually the cluster head can build the topology. Also the cluster heads (IMs) send their topology information to the CM, if there is any change or periodically to keep the topology updated. However, to keep the number of this transmission as minimum as possible, when a cluster head forwards the topology information of others, then it includes its own topology information if there is any changes.

The sink node (so the CM) accumulate the received topology information from the cluster head and generates the topology of the network. Note that, the neighbor list of normal sensor nodes should include nodes from neighboring cluster otherwise the topology information will be incomplete. As shown in Fig. 2(b), sensor nodes n_2 and n_3 have neighbors from both the two clusters. Therefore the CM can generate the complete topology. The topology generation algorithm can be summarized as follows:

[Phase 1]

1. Once a node selects it cluster head, it sends a joining message with its neighbor list.
2. Once a node hears transmission from any new node, it includes that node in its neighbor list.
3. Whenever any node includes a neighbor in the neighbor list, in the next data packet it piggybacks the inclusion of a new neighbor with id.

[Phase 2]

1. The IM accumulates the neighbor lists of all the members, and sends the topology information to the CM.
2. The CM accumulates the data receives from the IMs and generate the complete topology.
3. To keep the list updated, the IMs send their topology periodically. Whenever any IM forwards the cluster update message of other cluster head it includes its own topology if there is any change. This keeps the number of transmission a minimum number.

4.2.3 Cluster Reformation

A cluster head usually consumes more energy than a normal sensor node since it has forward the data of not only the members but also the other cluster heads. Therefore for balance energy consumption, it is necessary to change the cluster head periodically, a mechanism known as cluster reformation. However if the clusters are reconfigured by flooding as the initial cluster creation process, then such a re-clustering mechanism is very costly for the sensor network in terms of energy consumption. Furthermore during the cluster reformation process, usual data transfer may not be possible and so cluster reconfiguration process should be low cost, light weight and smooth.

The CM has the complete topology information and by analyzing the received packet it can find the approximate level of residual energy for each sensor node. Also, for better energy knowledge, every node can periodically inform the energy level but it is very costly. Now based on the topology and energy level of all the sensor nodes, the CM can execute the cluster creation mechanism ad find the minimum number of cluster heads and gateway nodes. Also it can select the member of a particular cluster. At the same time the CM can create the sleep/awake period of the sensor nodes, if necessary. Then it can send the reconfigured topology information to the existing cluster heads. Finally, each existing cluster head announces the new cluster head and their members and gateway node. This process does not require any flooding and the sink node with its powerful processing capability can generate minimum number of cluster heads and gateway nodes.

4.3 Fault Management

Since the wireless sensor networks are usually kept unattended, an efficient fault management system for these networks is very necessary. The purpose of a fault management system is to detect the fault and if possible recover from the faulty state of the network. For example, in hierarchical sensor network if a cluster head or gateway node gets faulty, both the fault detection (identify the faulty cluster head and gateway) and recovery (replace the node) is equally important. It is also noteworthy that, a fault management system should keep the probability of false alarm as minimum as possible.

4.3.1 Fault Detection

Fault detection is the process by which the network manager identifies a node which is malfunctioning or near to be dead and unable to sense or transmit/forward/receive data. If a normal sensor node dies then it will not create much problem, even though reliability may be decreased. But if a cluster head or a gateway node dies, then at least the data of that cluster will be lost and in the worst case such a failure may introduce network partition in the system.

In traditional IP networks, usual way to know whether a node is working properly or not is to get periodic keep-alive message from that node. But for sensor network such message exchange is very costly. So, fault detection operation in WSN should be light weight and done using passive information as much as possible. Our fault detection system works both locally by intermediate manager and centrally by a central manager.

The fault of normal sensors is detected by the intermediate manager; that is by the cluster head. If the sensors are supposed to send data periodically, then by analyzing the packets the cluster head can know which sensor is not responding. There is a chance that even the sensor is sending data the cluster head is not receiving the data due to collision. And repeated collision may create the CW so high that the cluster will not receive data from the sensor for a long time, even the sensor is working properly.

Inside the cluster each sensor maintains the state of its neighbors. If a sensor does not hear from any of its neighbor for a certain period of time then it informs the cluster head about that particular sensor. Cluster head and the neighbors maintain a timer T_1 for each of the neighbor sensors. If the cluster head or the neighbors hear a transmission from that sensor then they reset the timer. If the timer of the cluster head expires, then it waits before declaring the alarm. On the other hand, if the timer of the neighbor expires then in the next data packet, they piggyback that information. If the cluster head receives packets from any of the neighbors of that node, without any negative result then it again wait another random time. If no positive response before the timer expires or extends the random delay for three times then it generated an alarm and finally decide that the node is dead and inform the manager accordingly.

Now, for event driven sensor network, the sensor send periodic keep-alive message to the sink in the absence of event. This message may collide and cluster head may unable to receive the message. However some neighbors may receive the message. So, when a sensor sends a keep-alive message, they send the neighbor list and the last time when they hear from that node. Cluster head usage timer T_1 and reset it when

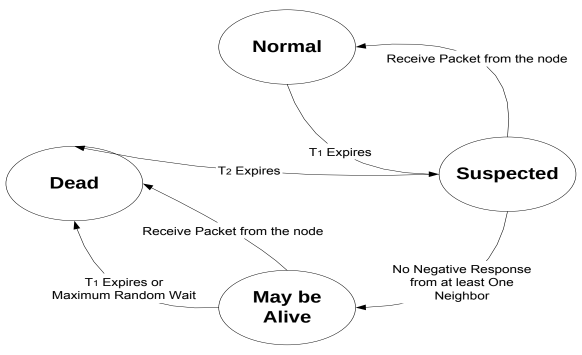
Normal
Receive Packet from the node
T1 Expires
Dead
T2 Expires
Suspected
Receive Packet from the node
T1 Expires or Maximum Random Wait
May be Alive
No Negative Response from at least One Neighbor

Figure 3. WSNMP architecture and relationship among different components

hear from a node or their neighbors. If timer expires then generates the alarm.

Fault detection of cluster head is more important than a normal senor. In case of periodic traffic, the central manager analyzes the packet received by the sink. As the central manager knows the topology of the network, it knows path of each cluster head to the sink. It again maintains two timers (T_1 and T_2) for each cluster heads and gateway nodes. When the sink receives a packet from that node or through that node, the central manager restarts the timer. If the timer expires, then the central manager suspects that node as dead. As the fault should be detected immediately, the value of T_1 should not be very high. When timer expires cluster head sends query packet to the node and wait another time T_2. If no response is received then it decides that the node is dead.

In event driven sensor network, in absence of events, the cluster head or gateway sends periodic message and cluster head usages the same timer mechanism to detect fault.

4.3.2 Fault Recovery

Once the fault is identified by the IM or CM, the next step is to recover the fault. If a normal sensor is died, then fault recovery is not very important. However fault recovery for the cluster head and gateway node is very important for the proper functioning and reliability of the sensor network.

If a cluster head is affected by the fault then the recovery mechanism has to select a new cluster head, and update the previous path and possibly may need to create a path for the new cluster head. Also, some of the cluster member of the old cluster head may not be within the transmission range of the new cluster head. Since the central manager (CM) has the topology of the whole network, the central manger should select a new cluster head with the minimum change in the existing infrastructure. Let us consider the following scenario in Fig. 4.

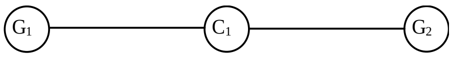

Figure 4. WSNMP architecture and relationship among different components

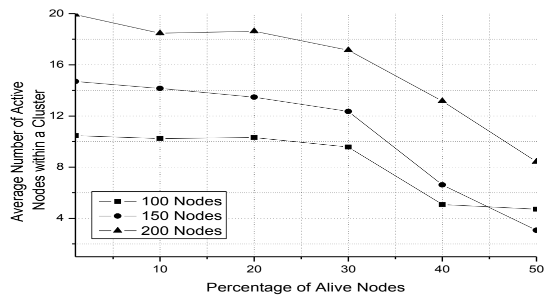

Figure 5. Average number of active nodes within a cluster vs. percentage of alive node

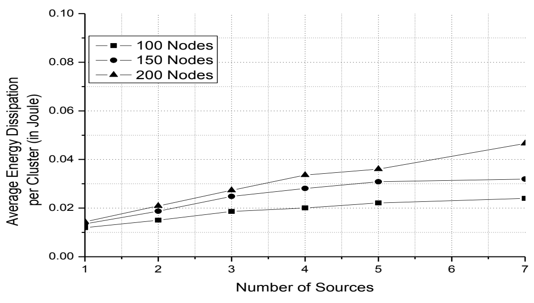

Figure 6. Average energy dissipation per cluster for increasing number of sources

If cluster head C_1 is dead, then the CM has to find a common node that is neighbor of both G_1 and G_2. Then the CM does not need to change the path. However if there is no node with sufficient energy then the CM can just select one suitable node as cluster head and add an additional gateway node that will connect the cluster head with either one of the two gateways as the other gateway is neighbor of the new cluster head. Then the CM sends a cluster head update and/or route update messages to the appropriate node(s). Also, if there are more than two neighboring gateway nodes then the CM has to reconstruct all the other paths as well and send appropriate route update messages.

In case of a faulty gateway node, the situation is reverse. The CM has to just reconstruct all the upstream paths. This may require selecting one or more new gateway nodes. After selecting the gateway nodes, the CM just sends route update message to the new gateways.

4.4 Performance Management

The performance management of WSNSM, monitors the performance of the network and keeps resource consumption as minimum as possible especially the energy uses. One of the major performance issues of the WSN is the event reliability which is defined as the number of unique data packets received by the sink node. For the optimum performance the management system sets the data generation rate of the sensors and also may keep some nodes in the sleep state and others in the normal live state.

4.4.1 Event Reliability

The event reliability depends on the proper detection of an event. The CM analyzes the packet receives by the sink and measures the event reliability periodically. If in a particular portion of the network the event state does not change so much then CM sends message to the cluster head(s) to reduce the data generation rate. Therefore it can allow more traffic from other regions at the converged network area if necessary. Also based on the event detection states the CM can select the active number of nodes for a particular cluster and accordingly inform the cluster heads.

4.4.2 Energy Usage

Energy is the one of the most important resources of the sensor networks, and so it should be used in the best possible way. The transceiver of a sensor consumes the maximum energy. And most of the sensor network placed redundant nodes in the networked area where two nearby nodes may generate almost identical data. Therefore, sensor nodes should be scheduled in such a way, a minimum number of nodes are kept alive at a particular moment while maintaining the network coverage. As the CM knows the topology of the network, thus it can select the minimum number of nodes to sense the whole network area. While selecting the cluster head during cluster reformation the CM also select the nodes under a particular cluster head and sends the information to the cluster head. Accordingly the cluster heads schedule the member nodes alternatively thus keeping the energy usage at the minimum level.

5. Performance Evaluation

To evaluate the performance of our proposed schemes we have performed extensive simulations using ns-2. Our simulation scenario consists of 100~200 nodes distributed uniformly over an area of 200m X 200m. All the nodes have a transmission range of 30m and sensing range of 15m. Events are generated randomly and nodes within the event radius generate CBR traffic of rate 5 packets per second (pps) and 64 byte in size. Data rate for the network is 300 kbps and each node is equipped with 2 joule of energy initially. We have implemented our hierarchical management scheme and used following matrices are used to realize the performance of proposed schemes:

Network Coverage: It indicates the number of active of nodes for the defined transmission range of the network.

Energy Dissipation: It indicates the average energy expenditure by the cluster heads for a particular event's data dissemination.

The number of active nodes within the transmission range indicates reliability of the network. If the nodes are well distributed, mathematically 6 nodes are enough to cover a circular region. In Fig. 5 we plot the number of active nodes within a cluster for decreasing percentage of alive nodes in the network. The graph shows our management scheme exhibits 12~13 active nodes per cluster even the alive nodes reduce to 70%.

Figure 6 shows the amount of average energy dissipation by cluster under increasing number of sources. For a good management scheme the energy consumption should be balanced so that the lifetime of the network is increased. According to simulation the amount of energy dissipation increases with the increasing number of sources in the cluster. Also, for high density network the energy dissipation is little high as more nodes remain active.

6. Conclusion

In this paper, we proposed a wireless sensor network management protocol named WSNMP, which proposes a mechanism to generate the topology of the sensor network and based on this efficiently reconfigure the network periodically. Also we proposed a fault detection and recovery mechanism for sensor networks and finally addressed the performance management mechanism to get the optimum service with minimum energy consumptions.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci: Wireless Sensor Networks: A Survey, Computer Networks, vol. 38(4) 2002.

[2] L.B. Ruiz, J.M.S. Nogueria and A.A.F Loureiro: MANNA: a management architecture for wireless sensor network, IEEE communications magazine, Vol. 41, 2003.

[3] H. Song, D. Kim, K. Lee, and J. Sung: Upnp-Based Sensor Network Management Architecture, in Proc. ICMU Conf., 2005.

[4] N. Ramanathan, E. Kohler, and D. Estrin: Towards a Debugging System for Sensor Networks, International Journal for Network Management, vol. 15(4) 2005.

[5] 5. M. Turon: MOTE-VIEW: A Sensor Network Monitoring and Management Tool, in Proc. IEEE EmNetS-II Workshop, 2005.

[6] 6. N. Ramanathan and M. Yarvis: A Stream-oriented Power Management Protocol for Low Duty Cycle Sensor Network Applications, in Proc. IEEE EmNetS-II Workshop, 2005.

[7] A. Boulis and M.B. Srivastava: Node-level Energy Management for Sensor Networks in the Presence of Multiple Applications, in Proc. IEEE PerCom Conf., 2003.

[8] J. Zhang, E.C. Kulasekere, K. Premaratne, and P.H. Bauer: Resource Management of Task Oriented Distributed Sensor Networks, in Proc. IEEE ICASSP Conf., 2001.

[9] M. Perillo and W.B. Heinzelman: Providing Application QoS through Intelligent Sensor Management, in Proc. IEEE SNPA Conf., 2003.

[10] R. Tynan, D. Marsh, D. OKane, and G. M. P. OHare: Agents for Wireless Sensor Network Power Management, in Proc. IEEE ICPPW Conf., 2005.

[11] Z. Ying and X. Debao: Mobile Agent-based Policy Management for Wireless Sensor Networks, in Proc. IEEE WCNM Conf., 2005.

[12] W. Louis Lee, A. Datta, and R. Cardell-Oliver: WinMS: Wireless Sensor network-Management system, An Adaptive Policy-based Management for Wireless Sensor Networks, Tech. Rep. UWA-CSSE-06-001, The University of Western Australia, 2006.

[13] B. Deb, S. Bhatnagar, and B. Nath: A Topology Discovery Algorithm for Sensor Networks with Applications to Network Management," Tech. Rep. DCS-TR-441, Rutgers University, 2001.

[14] B. Deb, S. Bhatnagar, and B. Nath: STREAM: Sensor Topology Retrieval at Multiple Resolutions, Kluwer Journal of Telecommunications Systems, vol. 26(2) 2004.

[15] T.H. Kim and S. Hong: Sensor Network Management Protocol for State-Driven Execution Environment, in Proc. ICUC Conf., 2003.