

Autonomous Management of Large-Scale Ubiquitous Sensor Networks^{*}

Jong-Eon Lee¹, Si-Ho Cha^{2**}, Dae-Young Kim¹, and Kuk-Hyun Cho¹

¹ Department of Computer Science, Kwangwoon University, Korea
{jelee, dykim, khcho}@cs.kw.ac.kr

² Department of Computer Engineering, Sejong University, Korea
sihoc@sejong.ac.kr

Abstract. A framework for the autonomous management of large-scale ubiquitous sensor networks called SNOWMAN (SeNsOr netWork MANagement) is proposed in this paper. In large-scale ubiquitous sensor networks, a huge number of sensor nodes are deployed over a wide area and long distances and multi-hop communication is required between nodes. So managing numerous ubiquitous sensor nodes directly is very complex and is not efficient. The management of large-scale ubiquitous sensor networks therefore must be autonomic with a minimum of human interference, and robust to changes in network states. The SNOWMAN is responsible for monitoring and controlling ubiquitous sensor networks based on policy-based management paradigm. It allows administrators to simplify and automate the management of ubiquitous sensor networks. It can also reduce the costs of managing sensor nodes and of the communication among them using a new hierarchical clustering algorithm.

1 Introduction

A large-scale ubiquitous sensor network (USN) consists of a huge number of sensor nodes, which are tiny, low-cost, low-power radio devices dedicated to performing certain functions such as collecting various environmental data and sending them to sink nodes (or base stations). In this network, a large number of sensor nodes are deployed over a wide area and long distances and multi-hop communication is required between nodes and sensor nodes have the physical restrictions in particular energy and bandwidth restrictions. So managing numerous wireless sensor nodes directly is very complex and is not efficient. To intelligent autonomous management, sensor nodes should be organized and managed automatically and dynamic adjustments need to be done to handle changes in the environment. The autonomous management of large-scale USNs must be able to know the changes in networks and to deal with the changes in a minimum of human interference.

^{*} The present research has been conducted by the Research Grant of Kwangwoon University in 2005.

^{**} Corresponding Author

We therefore propose an autonomous management framework for large-scale USNs called SNOWMAN (SeNsOr netWork MANagement), which is based on policy-based management (PBM) [1] paradigm. It allows administrators to simplify and automate the management of ubiquitous sensor networks. The proposed framework can also reduce the costs of managing sensor nodes and of the communication among them using a new hierarchical clustering algorithm.

This paper is structured as follows. Section 2 investigates related researches. Section 3 discusses the architecture, components, and clustering algorithm of the SNOWMAN. Section 4 presents the implementation of the SNOWMAN and the testbed of a USN. Finally in section 5 we conclude the paper.

2 Backgrounds

There is few related researches on the management of sensor networks.

Linyer B. Ruiz designed the MANNA architecture [2] for wireless sensor network (WSN), which considers three management dimensions: functional areas, management levels, and WSN functionalities. He also proposed WSN models to guide the management activities and the use of correlation in the WSN management. However, he described only conceptual view of the distribution of management functionalities in the network among manager and agent. He also suggested both locations of manager and agents and functions they can execute.

Chien-An Lee [3] proposed an intelligent self-organization management mechanism for sensor networks. The nodes are classified into three levels according to their functionality. The nodes in the low level are managed by those in the higher level and form hierarchical management structures. His work indicates how high-level nodes form a cluster through a contest with low-level nodes.

However, all of the previous works did not cover the implementation of management systems. Comparing to the previous works, our SNOWMAN provides an efficient management mechanism by integrating the functions of PBM and the hierarchical clustering architecture.

Wireless & ubiquitous sensor networks have need data aggregation to reduce communication bandwidth and energy consumption. Considering this, it may be advantageous to organize the sensors into clusters. In the clustered environment, the data gathered by the sensors is communicated to the data processing center through a hierarchy of cluster heads. Nowadays, several clustering algorithms have been proposed. Noted two Schemes are LEACH and LEACH-C.

LEACH (Low Energy Adaptive Clustering Hierarchy) [4] includes distributed cluster formation, local processing to reduce global communication, and randomized rotation of the cluster heads. These features leads a balanced energy consumption of all nodes and hence to a longer lifetime of the network.

LEACH-C (LEACH-Centralized) [5] is use a central control algorithm to form the clusters may produce better clusters by dispersing the cluster head nodes throughout the network. This is the basis for LEACH-C, a protocol that uses a centralized clustering algorithm and the same steady-state protocol as LEACH. Therefore the base station determines cluster heads based on nodes' location

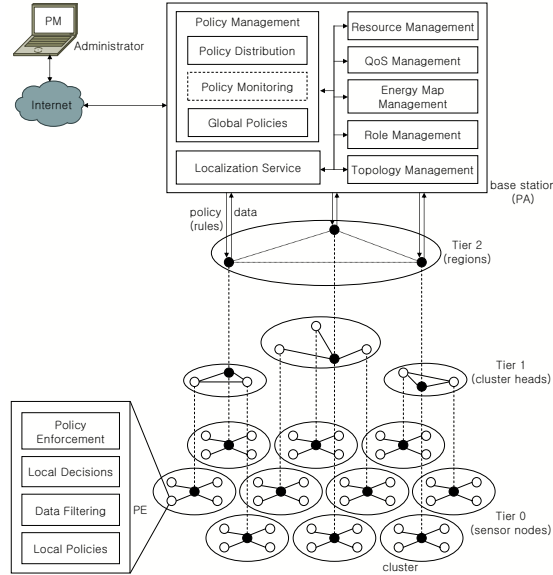


Fig. 1. SNOWMAN Framework for autonomous management

information and energy level. This feature leads to organize robust clustering topology. However, frequent communications between the base station and other sensor nodes increases communication cost.

To improve energy efficiency and consider management viewpoint, we introduce a hierarchical clustering scheme SNOW_{CLUSTER}.

3 SNOWMAN(SeNsOr netWork MANagement)

3.1 Overview of Architecture

To facilitate scalable and localizable management of sensor networks, SNOWMAN constructs 3 tier regional hierarchial cluster-based sensor network: regions, clusters, and sensor nodes as shown in Fig. 1.

In the architecture, a sensor network is comprised of a few regions and a region covers many clusters has several cluster head nodes. Sensor nodes should be aggregated to form clusters based on their power levels and proximity. In 3 tier regional hierarchial architecture of SNOWMAN, cluster heads constitute the routing infrastructure, and aggregate, fuse, and filter data from their neighboring common sensor nodes. The policy agent (PA) can deploy specific policies into particular areas (or clusters) to manage just singular regions or phenomena by more scalable manner. So, SNOWMAN framework is very useful to regionally manage the sensor networks.

The policy manager (PM) is used by an administrator to input different policies, and is located in a manager node. A policy in this context is a set of rules that assigns management actions to sensor node states. The PA and the policy enforcer (PE) reside in the base station and in the sensor node, respectively. The PA is responsible for interpreting the policies and sending them to the PE. The enforcement of rules on sensor nodes is handled by the PE. In a USN, individual nodes will not be able to maintain a global view of the network. Such a task is well suited for a machine not constrained by battery or memory. This is the reason for having the PA on the base station.

It is the job of the PA to maintain this global view, allowing it to react to larger scale changes in the network and install new policies to reallocate policies (rules). If node states are changed or the current state matches any rule, the PE performs the corresponding local decisions based on local rules rather than sends information to base station repeatedly. Such policy execution can be done efficiently with limited computing resources of the sensor node.

3.2 Functional Components

The PA consists of several functional components: policy distribution, policy monitoring, resource management, energy map management, QoS management, topology management, role management, and localization service. Localization service in the context implies the scalability of management to regionally manage the sensor networks. It is achieved via role management and topology management. Global policies are specified by a network administrator in a logically centralized fashion, and are expected to be static.

Policy distribution is the first essential task in ensuring that nodes are managed consistently with the defined policies. We design and implement a TinyCOPS-PR protocol that is similar to COPS-PR [6] protocol to deploy policies into sensor nodes. COPS-PR protocol is an extension for the COPS protocol to provide an efficient and reliable means of provisioning policies. The PA communicates with the PE using the TinyCOPS-PR protocol to policy distribution. TinyCOPS-PR allows asynchronous communication between the PA and the PEs, with notifications (reports, changes in policies, etc.) conveyed only when required.

However, to provide robust management of the network, it is desirable to have an independent policy monitoring process to ensure that the deployed policies behave well as defined in them. Though the policy monitoring is desirable, it is achieved via passive methods because of the resources of network are scarce. Energy map management continuously updates the residual energy levels of sensor nodes, especially of cluster heads and region nodes. This energy map management is also achieved via topology management process. Topology management consists of a topology discovery, resource discovery, and role discovery. Resource management and role management manage the detected resources and roles, respectively. QoS management is a part of policy management using QoS policies like bandwidth allocation for emergency. Energy map management and/or QoS management go through an aggregation and fusion phase when energy and/or

QoS information collected are merged and fused into energy and/or QoS contours by means of cluster heads.

The PE enforces local policies assigned by the PM to make local decisions and filter off unessential redundant sensed data. To do this, the PE consists of policy enforcement function, local decision function, data filtering function, and local policies. The PE communicates with the PA via TinyCOPS-PR protocol to be assigned local policies.

3.3 Hierarchical Clustering Algorithm: SNOW_{CLUSTER}

We propose a clustering scheme solely from a self-management viewpoint of WSN. Each sensor node autonomously elects cluster heads based on a probability that depends on its residual energy level. The role of a cluster head is rotated among nodes to achieve load balancing and prolong the lifetime of every individual sensor node. To do this, SNOWMAN re-clusters periodically to re-elect cluster heads that are richer in residual energy level, compared to the other nodes. We assume all sensor nodes are stationary, and have knowledge of their locations. Even though nodes are stationary, the topology may be dynamic because new nodes can be added to the network or existing nodes can become unavailable with faults and battery exhaustion.

SNOWMAN constructs hierarchical cluster-based sensor network using SNOW_{CLUSTER} clustering algorithm as seen in Table 1. SNOW_{CLUSTER} takes a couple of steps to accomplish the hierarchical clustering: 1) cluster head selection and 2) region node selection. In order to elect cluster heads, each node periodically broadcasts a discovery message that contains its node ID, its cluster ID, and its remaining energy level.

A node declares itself as a cluster head if it has the biggest residual energy level of all its neighbor nodes, breaking ties by node ID. Each node can independently make this decision based on exchanged discovery messages. Each node sets its cluster ID (*c_id*) to be the node ID (*n_id*) of its cluster head (*c_head*). If a node *i* hears a discovery message from a node *j* with a bigger residual energy level (*e_level*) than itself, node *i* sends a message to node *j* requesting to join the cluster of node *j*. If node *j* has already resigned as a cluster head itself, node *j* returns a rejection, otherwise node *j* returns a confirmation. When node *i* receives the confirmation, node *i* resigns as a cluster head and sets its cluster ID to node *j*'s node ID. After forming clusters, region nodes are elected from the cluster heads.

When the cluster head selection is completed, the entire network is divided into a number of clusters. A cluster is defined as a subset of nodes that are mutually reachable in at most 2 hops. A cluster can be viewed as a circle around the cluster head with the radius equal to the radio transmission range of the cluster head. Each cluster is identified by one cluster head, a node that can reach all nodes in the cluster in 1 hop.

After the cluster heads are selected, the policy agent (PA) should select the region nodes in the cluster heads. The PA receives cluster information messages (*c_info.msgs*) that contain cluster ID, the list of nodes in the cluster, residual

Table 1. SNOWCLUSTER Algorithm

```

// CLUSTER HEAD SELECTION
1.  $\forall x[\text{node}(x).\text{role} \leftarrow c.\text{head}]$ 
2.  $\forall x[\text{node}(x).\text{c\_id} \leftarrow \text{node}(x).\text{n\_id}]$ 
3.  $\forall x[\text{node}(x).\text{bcast}(\text{dis\_msg})]$ 
4. if  $\text{node}_i.\text{hears\_from}(\text{node}_j)$ 
5.     if  $\text{node}_i.\text{e\_level} < \text{node}_j.\text{e\_level}$ 
6.          $\text{node}_i.\text{req\_join}(\text{node}_j)$ 
7. if  $\text{node}_j.\text{role} \neq c.\text{head}$ 
8.      $\text{node}_j.\text{rej\_join}(\text{node}_j)$ 
9. else
10.     $\text{node}_j.\text{conf\_join}(\text{node}_j)$ 
11. if  $\text{node}_i.\text{rec\_conf}(\text{node}_j)$ 
12.     $\text{node}_i.\text{role} \leftarrow c.\text{member}, \text{node}_i.\text{c\_id} \leftarrow \text{node}_j.\text{n\_id}$ 

// REGION NODE SELECTION
1.  $\forall x[\text{if } \text{node}(x).\text{role} = c.\text{head}]$ 
2.     $\exists x[\text{node}(x).\text{bcast}(c.\text{info\_msg})]$ 
3. if  $PA.\text{rec}(c.\text{info\_msg})$ 
4.     $PA.\text{assign}(r.\text{node}), PA.\text{bcast}(r.\text{dec\_msg})$ 
5. if  $\text{node}_k.\text{rec}(r.\text{dec\_msg})$ 
6.    if  $\text{node}_k.\text{role} = c.\text{head}$ 
7.        if  $\text{node}_k.\text{n\_id} = r.\text{dec\_msg}.r.\text{id}$ 
8.             $\text{node}_k.\text{role} \leftarrow r.\text{node}, \text{node}_k.\text{r\_id} \leftarrow \text{node}_k.\text{n\_id}$ 
9.        else if  $\text{node}_k.\text{n\_id} \in r.\text{dec\_msg}.r.\text{list}$ 
10.             $\text{node}_k.\text{r\_id} \leftarrow r.\text{dec\_msg}.r.\text{id}$ 
11.             $\text{node}_k.\text{bcast}(r.\text{conf\_msg})$ 

```

energy level, and location data from all cluster heads. The PA suitably selects region nodes according to residual energy level and location data of cluster heads. If a cluster head k receives region decision messages ($r.\text{dec_msgs}$) from the PA, the node k compares its node ID with the region ID ($r.\text{id}$) from the messages. If the previous comparison is true, node k declares itself as a region node ($r.\text{node}$) and sets its region ID to its node ID. Otherwise, if node k 's node ID is included in a special region list ($r.\text{list}$) from the message, node k sets its region ID to a corresponding region ID of the message. The region node selection is completed with region confirmation messages ($r.\text{conf_msgs}$) broadcasted from all of cluster heads.

3.4 Evaluation of SNOWCLUSTER Algorithm

Simulation Environments In the experiment, ns-2 [7] network simulator was utilized and defined elements for establishing virtual experimental environment are as follows.

- Sensor network topology formed with each of 50, 100, 150, 200 nodes

- Sensor field with dimension of $100 * 100$
- Transmission speed of 1Mbps, Wireless transmission delay of 1ps
- Radio speed of $3 * 10^8$ m/s, Omni Antenna
- Lucent WaveLAN DSSS(Direct-Sequence Spread-Spectrum) wireless network interface of 914MHz
- Use of DSDV(Destination Sequenced Distance Vector) for routing protocol

Each experiment conducted on LEACH, LEACH-C, and SNOW_{CLUSTER}. In addition, management messages were applied for all cases and the processing power of sensor nodes were eliminated because it was insignificant compared to the amount of energy consumed in communications.

Energy Consumption during 10 Rounds based on the Number of Clusters Generated Fig. 2 is a graph that shows the generation of 1 to 10 clusters in a network topology formed with 100 sensor nodes for each clustering algorithm and one that shows the results of energy consumption measurement during 10 rounds based on the number of each cluster generated.

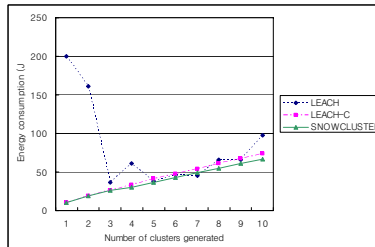


Fig. 2. Energy Consumption during 10 Rounds based on the Number of Clusters Generated

In case of LEACH, until the number of clusters generated is 2, it shows significantly higher energy consumption compared to the other clustering algorithms, but after generation of more than 3, it stabilized showing gradual increase. LEACH-C shows the progressive increase in energy consumption from round 1 to round 10. Similar to LEACH-C, SNOW_{CLUSTER} also shows the results of gradual increase, but its consumption rate is slightly less than that of LEACH-C. However, in the case of LEACH-C and SNOW_{CLUSTER}, due to the fact an unexpected increase in the number of cluster formation shows the increase in energy consumption, the most efficient number of clustering formation must be 1 from the perspective of energy consumption.

Energy Consumption during Transmission of Management Messages Fig. 3 is the result showing the amount of energy that is consumed during trans-

mission of management message from base station to sensor nodes after formation of three clusters in the network topology of 200 nodes.

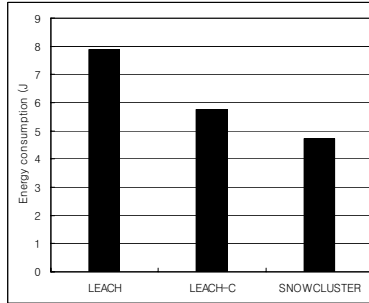


Fig. 3. Energy Consumption during Transmission of Management Messages from Base Station to Sensor Nodes

In case of LEACH, because it does not have the position information of the nodes, inefficient routing is being resulted, and as a result, significantly greater amount of energy is consumed in transmitting management messages. SNOWCLUSTER clustering algorithm showed a result of decrease in the amount of energy consumed in the transmission of message compared to LEACH-C, and this is because with the addition of region node selection process, it has a single region node that plays the role of primary message transmission compared to the three cluster heads for LEACH-C, and because it transmits messages using the remaining two cluster heads, total number of communication is reduced which results in reduction in communication expense.

4 Implementation

4.1 Testbed Network

Our current work has focused on validating some of our basic ideas by implementing components of our framework on Nano-24 [8] platform using the TinyOS programming suite.

The Nano-24 uses Chipcon CC4220 RF for transmission and support 2.4 Ghz, Zigbee. The sensor node uses atmega 128L CPU with 32KBytes main memory and 512 Kbytes flash memory. The Nano-24 also supports Qplus-N sensor network development environment that ETRI (Electronics and Telecommunications Research Institute) developed. We organized a testbed network was composed 10 Nano-24 nodes. Each node contains SNOWMAN's PE to support policy-based management as shown in Fig. 4. In this testbed, all sensor nodes are configured to hierarchical clustering architecture according to the SNOWCLUSTER clustering mechanism.

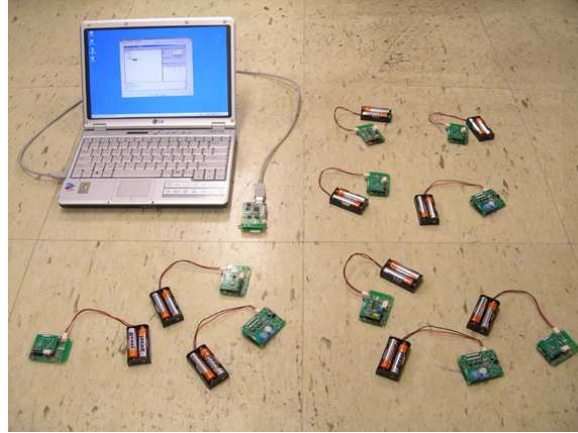


Fig. 4. Testbed Network for SNOWMAN

4.2 SNOWMAN

The PM and PA of SNOWMAN framework are implemented on Windows XP systems using pure JAVA. The PE is implemented on TinyOS in the Nano-24 nodes using gcc.

Fig. 5 shows the input forms for policy information on the PM. We use the XML technologies to define and handle global policies. There are several advantages of using XML in representing global policies [9]. Because XML offers many useful parsers and validators, the efforts needed for developing a policy-based management system can be reduced. To define XML policies, we customized and used the Scott's XML Editor [10]. The defined policies are stored locally in the policy storage of the PM and are stored remotely in the policy storage of the PA. PM communicates with PA via simple ftp for policy transmissions. To policy distribution to sensor nodes, we also design and implement TinyCOPS-PR that is simplified suitably for ubiquitous sensor networks.

5 Conclusion

In this paper, we proposed and implemented a autonomous management framework for large-scale ubiquitous sensor networks, called SNOWMAN. The SNOWMAN can reduce the costs of managing sensor nodes and of the communication among them using hierarchical clustering architecture. SNOWMAN can also provide administrators with a solution to simplify and automate the management of USNs using PBM paradigm. This paper also presented and evaluated the clustering algorithm, SNOWCLUSTER. In performance evaluation of clustering and managing ubiquitous sensor networks, SNOWCLUSTER showed more improvement than LEACH and LEACH-C.

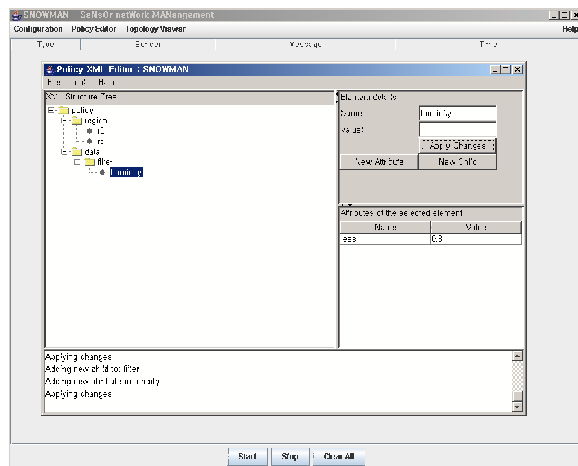


Fig. 5. Snapshot of SNOWMAN Policy Manager (PM)

We are currently at the stage of implementation of the business logic of SNOWMAN. We plan to experiment with and demonstrate the system on laboratory testbeds using Nano-24 sensor nodes.

References

1. R. Yavatkar, et al., A Framework for Policy-based Admission Control, IETF RFC 2753, January 2000.
2. Linnyer B. Ruiz, et al., MANNA: A Management Architecture for Wireless Sensor Networks, IEEE Communications Magazine, Volume 41, Issue 2, February 2003.
3. Chien-An Lee et al., Intelligent Self-Organization Management Mechanism for Wireless Sensor Networks, <http://www.ndhu.edu.tw/fdoffice/exchange/CYC-paper.pdf>.
4. W. Heinzelman, et al., Energy-Efficient Communication Protocol for Wireless Microsensor Networks, Proc. IEEE Int. Conf. System Sciences, vol. 8, January 2000.
5. W. Heinzelman, Application-Specific Protocol Architectures for Wireless Networks, PhD thesis, Massachusetts Inst. of Technology, June 2000.
6. K. Chen, et al., COPS usage for Policy Provisioning (COPS-PR), IETF RFC 3084, March 2001.
7. NS-2: <http://www.isi.edu/nsnam/ns/>
8. Nano-24: Sensor Network, Octacomm, Inc., <http://www.octacomm.net>
9. Si-Ho Cha, et al., A Policy-Based QoS Management Framework for Differentiated Services Networks, LNCS 2662, August 2003.
10. Scott Hurring, XML Editor, <http://hurring.com/code/java/xmleditor/>.