Multi-Agent System for Fault Tolerance in Wireless Sensor Networks

HwaMin Lee[1], Se Dong Min[2], Min-Hyung Choi[3] and DaeWon Lee[4,*]
[1] Department of Computer Software Engineering, Soonchunhyang University
Shinchang, Asan - Korea
[e-mail: leehm@sch.ac.kr]
[2] Department of Medical IT Engineering, Soonchunhyang University
Shinchang, Asan - Korea
[e-mail: sedongmin@sch.ac.kr]
[3] Department of Computer Science and Engineering, University of Colorado Denver
Denver - USA
[e-mail: sedongmin@sch.ac.kr]
[4] Division of General Education, SeoKyeong University
Seoul - Korea
[e-mail: daelee@skuniv.ac.kr]
*Corresponding author: DaeWon Lee

Received October 26, 2015; accepted December 9, 2015; published March 31, 2016

Abstract

Wireless sensor networks (WSN) are self-organized networks that typically consist of thousands of low-cost, low-powered sensor nodes. The reliability and availability of WSNs can be affected by faults, including those from radio interference, battery exhaustion, hardware and software failures, communication link errors, malicious attacks, and so on. Thus, we propose a novel multi-agent fault tolerant system for wireless sensor networks. Since a major requirement of WSNs is to reduce energy consumption, we use multi-agent and mobile agent configurations to manage WSNs that provide energy-efficient services. Mobile agent architecture have inherent advantages in that they provide energy awareness, scalability, reliability, and extensibility. Our multi-agent system consists of a resource manager, a fault tolerance manager and a load balancing manager, and we also propose fault-tolerant protocols that use multi-agent and mobile agent setups.

Keywords: fault tolerance, wireless sensor networks, multi-agent, mobile agent

A preliminary version of this paper was presented at APIC-IST 2015, and was selected as an outstanding paper. This research was supported by the Soonchunhyang University Research Fund and This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2014R1A1A2057878).

http://dx.doi.org/10.3837/tiis.2016.03.021 **ISSN : 1976-7277**

1. Introduction

Over the past decade, wireless sensor networks (WSNs) have gained attention from academia and industry around the world and have been implemented in a variety of applications, such as for environment monitoring, disaster management, equipment maintenance, automation and control, emergency response and ecophysiology [1, 2]. WSNs are self-organized networks that consist of a large number of distributed sensor devices that are deployed over a wide geographical area to monitor physical phenomena, including temperature, humidity, vibrations, etc. In WSNs, the sensor nodes have limited computational power and ability to communicate, and the communication bandwidth is also limited. A sensor node is a tiny device that includes three components: a processing unit to process local data, a sensing unit to acquire data from the physical environment, and a wireless transceiver that transfers the sensing information to the base station. These sensor nodes perform data sensing and processing tasks and also communicate with each other. In WSNs, sensor node batteries cannot be replaced or recharged, and applications and services should be designed in an energy efficient manner. In WSNs, a failure can occur as a result of various reasons. First, sensor nodes may fail due to battery exhaustion, hardware and software failures and malicious attacks. Second, communication links are failure-prone in WSNs [3, 4, 5], causing dynamic changes and network partitions in the sensor network. Since sensor nodes are prone to failure, a failure detection and fault recovery techniques is an essential requirements in WSNs. However, a base station that collects and identifies information from each of the sensor nodes to provide fault tolerance can be very expensive.

In this paper, we propose a novel fault-tolerant, multi-agent WSN system. Fault tolerance, load balancing, and energy consumption are managed in the WSNs with a resource manager, a fault tolerance manager and a load balancing manager. This system uses multi-agent and mobile agent configurations to manage WSNs and to balance loads across the network. The multi-agent system executes a predefined algorithm to find the resource status and the load at the sensor node and the base station. We use a mobile agent to aggregate and carry the sensing data. The mobile agent can reduce the data redundancy and the communications overhead, and as a result, the network exhibits less complexity and overhead during message transfer relative to existing systems. In addition, the network operates efficiently in terms of the energy consumption, fault tolerance, response time and network delay.

This paper is organized as follows. Section 2 reviews the terminology and presents a survey of the fault tolerance approaches in WSNs. We define a multi-agent system in Section 3 and propose fault tolerance and load balancing algorithms in Section 4. The results of the simulation are presented in Section 5. Finally, we conclude this paper and discuss future work in Section 6.

2. Related Work

2.1 Faults in the wireless sensor networks

To understand fault tolerant mechanisms, it is important to point out the differences between faults, errors, and failures [6, 7]. Any kind of fault causes an error, an error is an inaccurate system state, and such state may results in a failure. A failure is triggered by an error that

provides a wrong result according to the system specification, and so the system does not deliver the stated functionality.

Fig. 1 shows a layered classification of the components in a WSN that can cause faults [4]. A fault in each layer has the possibility to spread above each of the levels, and in this paper, we focus on the faults that can happen up to the sink at the sensor node.

1) *Node faults*: Nodes have several hardware and software components that can lead to trouble.

2) *Network faults*: Routing is a fundamental block of a WSN that is essential in the collection of sensor data, distribution of software and configuration updates, and coordination among the nodes.

3) *Sink faults*: For a higher level of the network, a sink device collects all of the data generated in the network and transmits it to the back-end system. It also is responsible for faults in its components. When a failure occurs at the sink when fault-tolerant measures have not been implemented, the aftereffect of the failure spreads to all nodes of network resulting in sensor nodes that cannot be accessed.

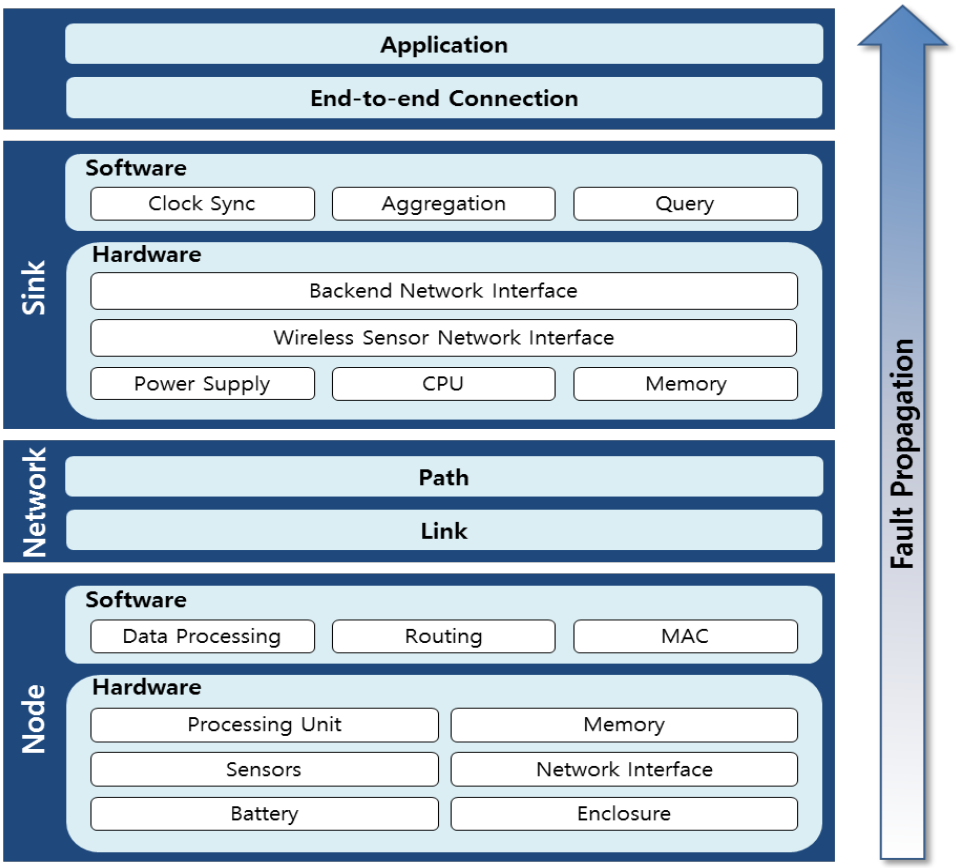

Fig. 1. Fault propagation in WSNs.

In this paper, the failures are classified as crash, omission, timing, value and arbitrary.

1) *Crash or omission*: An omission failure is that where a node that does not respond to a request. It leads to a crash failure when the service stops sending responses to any request.

2) *Timing*: Timing failures will only occur when the application specifies timing constraints that result in a timeout when processing a request.

3) *Value*: A service always considers an incorrect value during a periodical response. The incorrect value causes a value failure.

4) *Arbitrary*: Arbitrary failures are sometimes referred to as byzantine failures, and these include all types of failures that cannot be classified as failures that have been previously described. In general, byzantine failures are caused by a malicious service that not only behaves erroneously, but also fails to behave consistently when interacting with other services and applications.

Fault tolerance mechanisms are usually classified into retransmission and replication mechanisms.

1) *Transmission:* The most popular fault tolerance mechanism is to retransmit data packets to the sink device. In order to achieve a more rapid recovery, one of multiple paths is selected according to the minimum hop count or the minimum energy consumption. However, this method has some drawbacks in that it increases the network traffic and demands more resources. Also, the delivery delay to transmit an acknowledgment message can increase, and packet loss is caused by network collisions. Furthermore, the sensor nodes need a larger memory space to buffer the packet until an acknowledgment message arrives.

2) *Replication:* To reduce the packet delivery cost, a replication mechanism is used to provide fault tolerant routing protocols for the WSNs. In order to overcome path failures, a replication mechanism transmits multiple copies of the same packet over to different paths to ensure delivery of the original packet to the sink. The major drawbacks of using the replication mechanism are its high overhead when many packets are transmitted from each node to the sink device, the complex path of each of the nodes, and not being adaptive to errors in the communication channel.

2.2. Fault tolerance in wireless sensor networks

Ref. [7] classifies the fault management process into fault detection, diagnosis, and recovery. First of all, a fault is detected, and fault recovery techniques are carried out. Therefore, fault detection is the most important phase to provide an optimal fault tolerance for WSNs.

Several clustering techniques were proposed in Ref. [8,9]. For example, Low-Energy Adaptive Clustering Hierarchy (LEACH) [8] uses clustering to extend the network lifetime by applying a specific data dissemination protocol. In Ref. [9], a fully distributed clustering (HEED) algorithm was proposed to consider the remaining energy to select the CH of some candidates. However, the major drawback of HEED is that it requires more energy to communicate with the CH.

Several fault detection and fault tolerant techniques were proposed in Refs. [10-14]. In Ref. [10], a distributed fault detection algorithm was proposed. This algorithm identifies faulty nodes by conducting a comparison between neighbor nodes, and the decision made at each node is disseminated. Sympathy [11] uses a message-flooding approach to pool event data and current states from the nodes by using a broadcasting mechanism with many communication signals. However, this mechanism doesn't take the energy problem into account. In Ref. [12], a group-based algorithm was proposed where the same region should have similar values unless a node is at the boundary of the event-region. A fault-tolerant event boundary detection scheme that uses collaborative endorsement was proposed in Ref. [13]. To increase the resilience against the node compromise, the proposed algorithm allows multiple nodes to

collectively endorse a valid boundary. In Ref. [14], a distributed fault identification protocol was proposed. The Dynamic-DSDP protocol uses a comparison approach and a solution for a self-diagnosed problem.

In Refs. [15, 16], a cluster-based failure detection mechanism was proposed on a wireless ad hoc network environment. The most significant parameter of a WSN is its energy consumption. The WSN environment specification is different from that of an ad-hoc network environment. WSNs are power-constrained, and such approaches may not be suited for use with WSNs in an ad-hoc network. ZFTMA was proposed in Ref. [17] as a cluster-based fault tolerant management architecture. In this protocol, the entire network is divided into four zones and is managed with a hierarchical management scheme.

In Ref. [18], WFDA was proposed to provide fault detection in the network using a Wiener filter-based agent. The mobile agents are used to migrate from one node to another. In Ref. [19], DFDNM was proposed as a form of distributed fault-detection and node management scheme for use in cellular automata in wireless sensor networks. In DFDNM, the neighbor nodes detect a fault and verify the correlation of the sensing information that is collected.

3. Multi-Agent System

We propose a multi-agent system that provides resource management, fault tolerance, and load balancing for WSNs. We assume that the sensor nodes are stationary and are power constrained. A base station uses a multi agent system to ensure energy efficiency, fault tolerance and load balancing for WSNs, and it also creates a multi-agent setup to perform predefined tasks in the present cluster. Our multi-agent system consists of a resource manager, a fault tolerance manager and a load balancing manager. **Fig. 2** shows the components of the multi-agent system. Our system can provide energy efficient services through the use of a mobile agent. It supports code mobility over the base station and the sensor node, and each agent executes a predefined role for WSN management and shares information with the other nodes by using the communication interface.

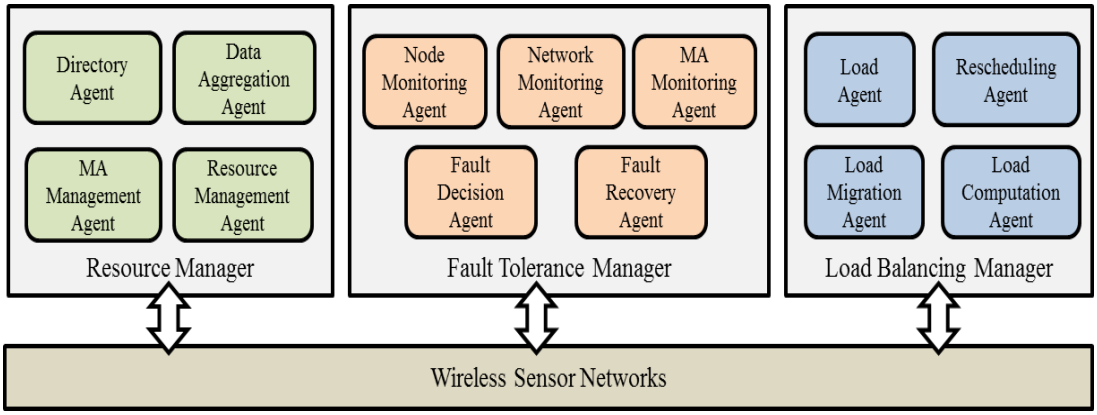

Fig. 2. Components of our multi-agent system.

3.1. Resource Manager

The main task of resource manager is to manage the agents, sensor nodes and base station. Our resource manager consists of a directory agent, a data aggregation agent, a mobile agent (MA) management agent and a resource management agent.

■ Directory agent

The directory agent manages meta information of the WSN and sensor nodes and provides meta information to service applications. The meta information consists of static information and dynamic information. The static information consists of the ID of the WSNs, ID and number of sensor nodes, type of sensor and actuator, processing specification of the sensor nodes. The dynamic meta information consists of the remaining battery and operation status of the sensor nodes, communication status of the WSNs, etc.

■ Data aggregation agent

The data aggregation agent gathers and aggregates data from multiple sensors and transmits the aggregated data to a base station and a cluster head node for job processing.

■ MA management agent

The MA management agent manages the creation, management and deletion of a mobile agent. It is inefficient for all sensors to transmit the sensed data directly to the base station because data generated from the neighboring sensor nodes is highly correlated and redundant. Therefore, we use a mobile agent to gather and aggregate data.

■ Resource management agent

The resource management agent performs resource discovery, resource selection and resource allocation requests. It makes an entry list for the processing tasks and makes requests to allocate resources to the sensor nodes. It selects the routing path and updates the network topology dynamically to conduct resource selection and resource allocation.

3.2 Fault Tolerance Manager

A fault tolerance manager is responsible to monitor the resources, detect failure, and engage in fault recovery. These are briefly introduced below. Our fault tolerance manager consists of a node monitoring agent, a network monitoring agent, an MA monitoring agent, a fault decision agent, and a fault recovery agent.

■ Node monitoring agent

The node monitoring agent monitors for a faulty state, normal state, and potential failure in the sensor node and the cluster head. It monitors the hardware fault-sensor, processing unit, memory, battery, and radio of a sensor node.

■ Network monitoring agent

The network monitoring agent monitors the communication bandwidth, congestion level, network disconnection, changes in topology, communication latency, and partition between the nodes. To track the link quality, we measure the percentage of the errorless packet that has been received.

■ MA monitoring agent

The MA monitoring agent monitors the state of the mobile agents and classifies the state into a processing state, a stop state, and an unknown state.

■ Fault decision agent

The fault decision agent determines that a failure has occurred by analyzing the state information of the node and network monitoring agent and identifying the node failure, potential failure, or network failure. We define the potential failure for a sensor node with a low battery that can enter a sleep state before completely shutting down. To enable automatic fault detection, we use a self-diagnosis method, group detection and hierarchical detection.

■ Fault recovery agent

The fault recovery agent carries out predefined fault tolerance techniques if the fault decision agent determines that a failure occurred. We use component replication and migration techniques to conduct the fault recovery.

3.3 Load Balancing Manager

In WSNs, sensors are constrained in terms of their battery life, size and communication range. As a result, they require multi-hop wireless communication in order to forward the data. Load balancing increases the lifetime of the sensor by reducing the battery consumption. We propose a load-balancing manager to balance the load across the network. Our load-balancing manager consists of a load agent, a load gathering agent, a load computation agent and a load migration agent.

■ Load agent

The load agent performs calculations for the resource load of a sensor node and a base station. The CPU load is the sum of the remaining processing time for the tasks running on the sensor node. The memory load is the sum of the page fault time for the tasks on the sensor node. The load of the I/O is the sum of the I/O processing time of the tasks on a sensor node. The total load of the sensor node is the predefined load calculation function.

■ Load gathering agent

The load gathering agent is a mobile agent and is responsible for the collection of load information from the sensor nodes, cluster head node and base station. It travels in the cluster region and disseminates the information that has been gathered to the base stations.

■ Load computation agent

The load computation agent computes the cost of sending messages to another sensor node and the base station. The cost is measured as the number of hops traveled by the message and the bandwidth lost due to the presence of selfish sensor nodes in the path. In a WSN, the battery power of a sensor node is a critical system resource, so this agent plays an important role in computing the cost in terms of the bandwidth loss. This agent then executes the cost computation policy.

■ Load migration agent

The load migration agent performs the task migration from a heavily loaded node to a lightly loaded node. It requests a rescheduling agent to decide whether or not the load migration has occurred. If it receives a rescheduling result from the rescheduling agent, it requests the allocation of new selected nodes to the resource management agent.

■ Rescheduling agent

The rescheduling agent evaluates the benefits that can be obtained from migrating the load and decides whether or not load migration occurs. The rescheduling agent also decides a new node list for a given task.

4. Fault Tolerance Protocol

Fault tolerance and load balancing are supported in a WSN, through the use of a novel multi-agent based system, which is referred to as a Multi-agent based Fault Tolerance System (MFTS). **Fig. 3** shows the architecture for our multi-agent based fault tolerance system. We use a cluster head technique with a Low-Energy Adaptive Clustering Hierarchy (LEACH) [8]. Our system provides energy efficient computations through the use of a multi-agent and mobile agent configuration.

The fault tolerance protocol of our MFTS is as follows. **Fig. 3** illustrates the steps taken by MAS_i to detect a failure in mobile agent A when MAS_{i-1} fails to receive the registration message M_{reg}^{i} and the deregistration message M_{dereg}^{i}.

1. When a mobile agent A arrives at a $processing\_node_i$ it registers with the message M_{reg}^{i} and FRS_{i-1} and receives its charge at the $processing\_node_{i-1}$ from the previous node's MAS running at the cluster head of a subnetwork MAS_{i-2}. Then A sends message M_{reg}^{i} to MAS_{i-1}.
2. MAS_{i-1} waits for the message M_{reg}^{i} for a given timeout interval, and if the timeout interval is reached, it requests MAS_i to determine the status of the mobile agent A.
3. MAS_i searches the database for the registration information M_{reg}^{i} when it receives requests from MAS_{i-1}.
4. If M_{reg}^{i} is found, MAS_i retransmits M_{reg}^{i} in order to recover the lost message. If M_{reg}^{i} is not found, MAS_i recovers the mobile agent A in $_i$ by asking MAS_{i-1} for the checkpoint. If $processing\_node_{i-1}$ fails to provide the message, MAS_i requests for the message from the MAS running at base station in the WSN. After the failure recovery procedure is completed, MAS_i retransmits the message M_{reg}^{i}.

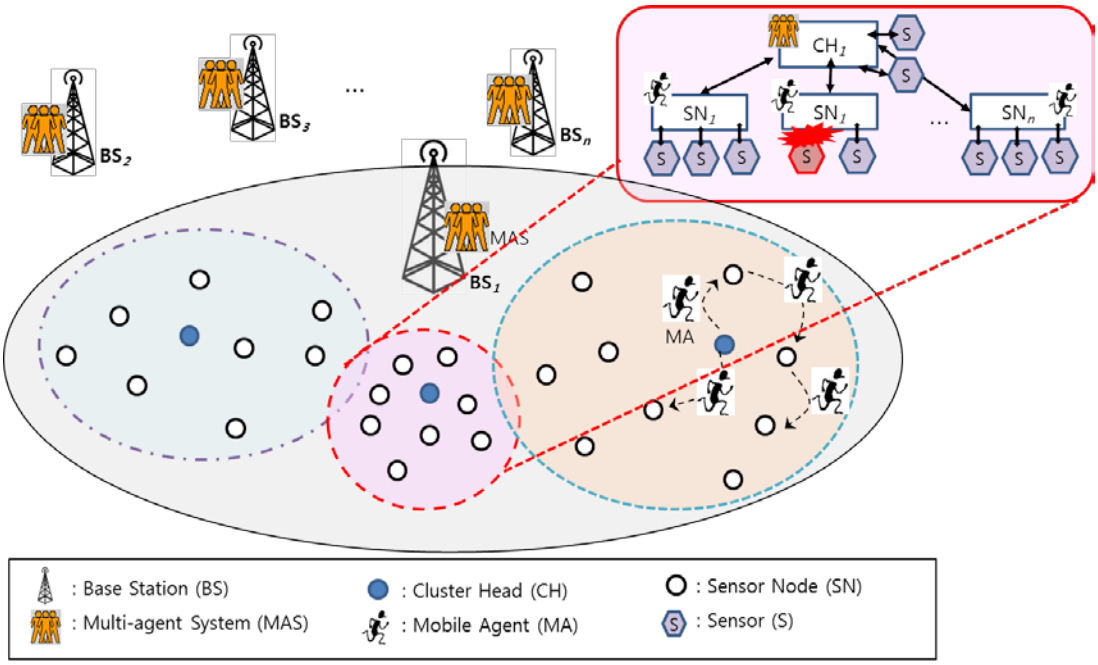

Fig. 3. Architecture of the multi-agent based fault tolerance system.

5. The mobile agent *A* performs the task at the *processing_node*$_i$ and conducts a checkpoint of the status at the *processing_node*$_i$. Then, it registers the message M_{dereg}^{i} at the *processing_ node*$_i$.
6. If the checkpoint fails, the process recruits the help of the MA*S* running at the cluster head of a subnetwork, and also recruits the help of the MA*S* at the base station of the network. If it fails again, it aborts the entire task.
7. Before the mobile agent *A* migrates from the *processing_ node*$_i$, it sends M_{dereg}^{i} to FRS_{i-1}. Upon receiving M_{dereg}^{i}, FRS_{i-1} gives charge of the mobile agent *A* to FRS_i at the *processing_node*$_i$.

5. Simulation

We implemented the MFTS in JAVA to conduct the simulation. **Fig. 4** shows the architecture of the system used in the simulation. Our simulation system consists of a sensor node, cluster head, base station, resource manager (RM), fault tolerance manager (FTM), load balancing manager (LBM) and simulator. We assume that the network size is 1 *Km* * 1 *Km*, the transmission ranger of the sensor node is 50 *m* and the network bandwidth is 1 *Mpbs*. For the simulation, our simulator randomly kills the sensors and the nodes.

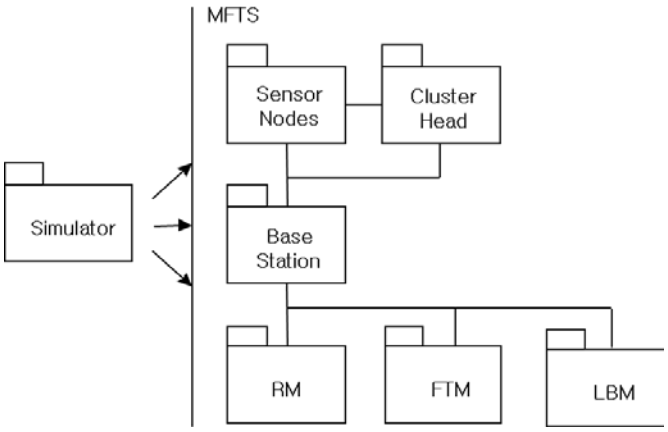

Fig. 4. Architecture of our simulation system.

We compared our MFTS to Wiener agent based fault detection [18] and a distributed fault detection and node management scheme (DFDNM) [19]. We measured the itinerary completion rate of the mobile agents in the sensor nodes, cluster head and base station. Our simulator kills sensor nodes randomly with an MTTF = 2000 *ms*.

Fig. 5 shows a comparison of the itinerary completion rate of the mobile agent for three different approaches to fault recovery, i.e. MFTS, Wiener and DFDNM. In **Fig. 5**, the itinerary completion rate of the mobile agent is better than that achieved with Wiener or DFDNM. The advantage of our MFTS is shown to be more significant as the number of nodes increases.

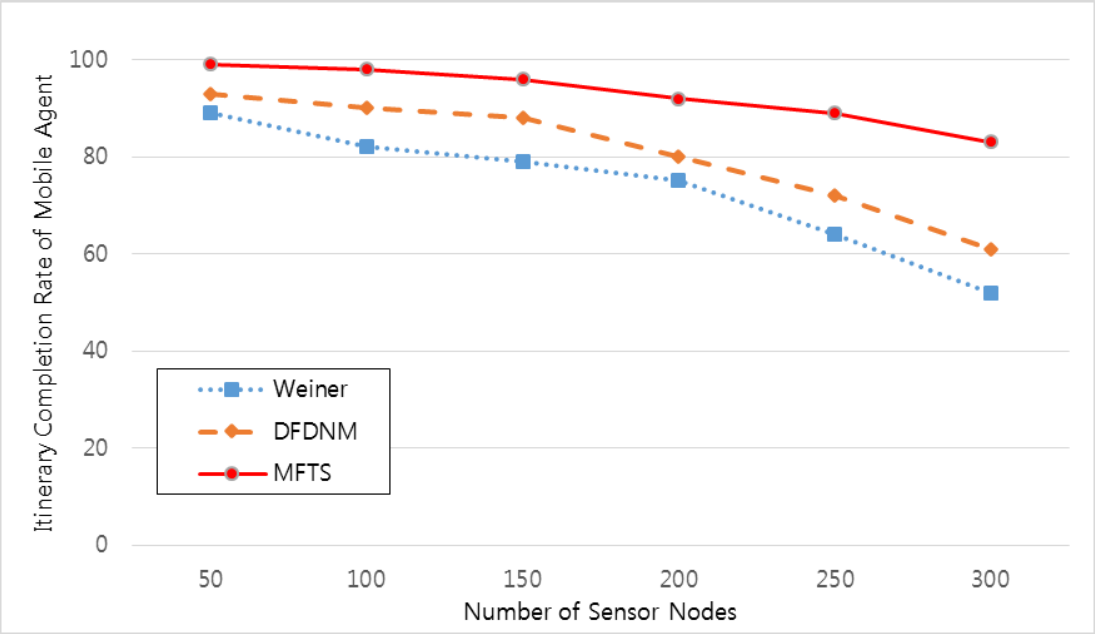

Fig. 5. Itinerary completion rate of the mobile agent.

5. Conclusion

A wireless sensor network consists of a large number of sensor nodes. WSN systems are characterized by their energy consumption, limited processing power, memory and bandwidth, scalability, fault tolerance, and adaptability. In WSNs, the sensors are prone to failure as a result of battery depletion, communication failure, limited resources, hardware failure, etc. Therefore, fault tolerance and energy efficiency are very important factors for WSNs. We propose a new multi-agent system framework that provides reliable, efficient operation, and we refer to this as a multi-agent based fault tolerance system (MFTS) and fault tolerance protocol. We designed a resource manager, a fault tolerance manager and a load balancing manager that provide fault tolerance and load balancing. We tested our system and proposed protocol by conducting a simulation and obtained promising experimental results. In the future, we plan to implement our multi-agent based fault tolerance system in a real WSN and will conduct a variety of experiments to measure the efficiency of the fault manager and the load balancing manager.

References

[1] Akyildiz I.F., Su W., Sankarasubramaniam Y. and Cayirci E., “Wireless sensor networks: a survey,” *Computer Networks (Elsevier)*, Vol. 38, no4, pp 393-422, March 2002. Article (CrossRef Link)

[2] A. Woo, T. Tong, and D. Culler, “Taming the underlying challenges of reliable multihop routing in sensor networks,” in *Proc. of ACM Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys'03)*, pp.14-27, November 2003. Article (CrossRef Link)

[3] Dong Woo Ryu, Kyung Jin Kang, Sang-Soo Yeo and Sang Oh Park, “Generating knowledge for the identification of device failure causes and the prediction of the times-to-failure in u-Healthcare environments,” *Personal and Ubiquitous Computing*, Springer, Vol. 17, Issue 7, pp. 1383-1394, October 2013. Article (CrossRef Link)
[4] L. Paradis and Q. Han, “A Survey of Fault Management inWireless Sensor Networks,” *Journal of Network and Systems Managemen*, Vol. 15, No. 2, pp.171-190, 2007. Article (CrossRef Link)
[5] Binod Vaidya, Dimitrios Makrakis, Jong Hyuk Park and Sang-Soo Yeo, "Resilient Security Mechanism for Wireless Ad hoc Network,” *Wireless Personal Communications*, Springer, Vol. 56, No. 3, pp. 385-401, February 2011. Article (CrossRef Link)
[6] H.M. Ammari and S.K. Das, “Fault tolerance measures for large-scale wireless sensor networks,” *Journal ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, Vol. 4, No. 1, pp.1-28, 2009. Article (CrossRef Link)
[7] YU, Mengjie; MOKHTAR, Hala; MERABTI, Madjid, “Fault management in wireless sensor networks,” *Wireless Communications, IEEE*, 14.6: 13-19, 2007. Article (CrossRef Link)
[8] W. B. Heinzelman, A. P. Chandrakasan, H. Balakrishnan, “An application-specific protocol architecture for wireless microsensor networks,” *IEEE Transactions on Wireless Communications*, vol. 1, pp.660-670, 2002. Article (CrossRef Link)
[9] O. Younis, S. Fahmy, “A Hybrid Energy Efficent Distributed Clustering Approach for Ad_Hoc sensor Network,” *IEEE Transcation on Mobile Computing*, 3(4):660-668, 2004. Article (CrossRef Link)
[10] LEE, Myeong-Hyeon; CHOI and Yoon-Hwa, “Fault detection of wireless sensor networks,” *Computer Communications*, 31.14: 3469-3475, 2008. Article (CrossRef Link)
[11] N. Ramanathan et al., "Sympathy for the Sensor Network Debugger," *ACM SenSys '05*, San Diego, CA, 2005. Article (CrossRef Link)
[12] B. Krishnamachari and S. Iyengar, “Distributed Bayesian Algorithms for Fault-Tolerant Event Region Detection in Wireless Sensor Networks,” *IEEE Transactions on Computers*, 53:241–250,March 2004. Article (CrossRef Link)
[13] K. Ren, K. Zeng and W. Lou, “Secure and Fault-Tolerant Event Boundary Detection in Wireless S ensor Networks,” *IEEE Transactions on Wireless Communications*, Vol. 7, No. 1, 2008. Article (CrossRef Link)
[14] ELHADEF, Mourad; BOUKERCHE, Azzedine; and ELKADIKI, Hisham, “A distributed fault identification protocol for wireless and mobile ad hoc networks,” *Journal of Parallel and Distributed Computing*, 68.3: 321-335, 2008. Article (CrossRef Link)
[15] A. T. Tai, K. S. Tso and W. H. Sanders, "Cluster-Based Failure Detection Service for Large-Scale Ad Hoc Wireless Network Applications,” *Dsensor node '04*, Florence, Italy, 2004. Article (CrossRef Link)
[16] P. Khilar, J. Singh and S. Mahapatra, “Design and Evaluation of a Failure Detection Algorithm for Large Scale Ad Hoc Networks Using Cluster Based Approach,” in *Proc. of IEEE, International Conference on Information Technology*, pp. 153-158, 2008. Article (CrossRef Link)
[17] Khan, Muhammad Zahid et al, "A fault-tolerant network management architecture for wireless sensor networks," in *Proc. of 11th Annual postgraduate Symposium on the Convergence of Telecommunication, Networking and Broadcasting*, ISBN. 2010. Article (CrossRef Link)
[18] M. Al-Kasassbeh and M. Adda, "Network fault detection with Wiener filter-based agent," *Journal of Network and Computer Applications*, vol. 32, pp. 824-833, 2009. Article (CrossRef Link)
[19] Indrajit Banerjee, Prasenjit Chanak, Biplab Kumar Sikdar and Hafizur Rahaman, "DFDNM: A Di stributed Fault Detection and Node Management Scheme for Wireless Sensor Network," *Commu nications in Computer and Information Science*, Vol. 192, pp 68-81, 2011. Article (CrossRef Link)

HwaMin Lee is a professor in the Department of Computer Software Engineering at Soonchunhyang University. She received her B.S., the M.S. and the Ph.D. degrees in Computer Science Education from Korea University in Seoul, Korea in 2000, 2002, and 2006, respectively. Her research interests include IoT, Cloud computing, Mobile computing, Grid computing, Wellness and resource management and fault tolerant system for large-scale distributed systems.

Se Dong Min was born in Seoul, Korea, on Dec. 23, 1975. He received MS and Ph.D. degree in electrical and electronic engineering from the department of Electrical and Electronics Engineering, Yonsei University, Seoul, Korea in 2010. He is at present working as an assistant professor in the Dept. of Medical IT Engineering of Soonchunhyang University, Korea. His research area includes biomedical signal instrumentation and processing, healthcare sensor application and mobile healthcare technologies.

Min-Hyung Choi received his M.S. and Ph.D. from University of Iowa in 1996 and 1999 respectively. He joined the Computer Science and Engineering Department at the University of Colorado at Denver in 1999. Currently he is the Director of Computer Graphics and Virtual Environments Laboratory. His research interests are in Computer Graphics, Scientific Visualization and Human Computer Interaction with an emphasis on physically-based modeling and simulation for medical and bioinformatics applications.

DaeWon Lee received his B.S. in division of Electricity and Electronic Engineering from Soonchunhyang University, Asan, Chung-Nam, Korea in 2001. He received his M.E. and Ph.D. degrees in Computer Science Education from Korea University, Seoul, Korea in 2003 and 2009. He is currently a full time lecturer in the Division of General Education at Seokyeong University in Korea. His research interests are in IoT, Mobile computing, Distributed computing, Cloud computing, and Fault-tolerant systems.