International Journal of Electrical and Computer Engineering (IJECE)
Vol. 10, No. 4, August 2020, pp. 4416~4425
ISSN: 2088-8708, DOI: 10.11591/ijece.v10i4.pp4416-4425

Self-checking method for fault tolerance solution in wireless sensor network

Muayad Sadik Croock, Saja Dhyaa Khuder, Zahraa Abbas Hassan
Computer Engineering Department, University of Technology, Iraq

Article Info

Article history:

Received Jan 6, 2020
Revised Mar 22, 2020
Accepted Mar 28, 2020

Keywords:

Fault tolerance
Self-checking technique
Software engineering
WSN

ABSTRACT

Recently, the wireless sensor network (WSN) has been considered in different application, particularly in emergency systems. Therefore, it is important to keep these networks in high reliability using software engineering techniques in the field of fault tolerance. This paper proposed a fault node detection method in WSN using the self-checking technique according to the rules of software engineering. Then, the detected faulted node is covered employing the reading of nearest neighbor nodes (sensors). In addition, the proposed method sends a message for maintenance to solve the fault. The proposed method can reduce the time between the detection and recovery of a fault to prevent the confusion of adopting wrong readings, in which the detection is making with mistake. Moreover, it guarantees the reliability of the WSN, in terms of operation and data transmission. The proposed method has been tested over different scenarios and the obtained results show the superior efficiency in terms of recovery, reliability, and continuous data transmission.

Corresponding Author:

Muayad Sadik Croock,
Department of Compute Engineering,
University of Technology,
Al-sinaa Street, Baghdad, Iraq.
Email: [1]Muayad.S.Croock@uotechnology.edu.iq; [2]120099@uotechnology.edu.iq;
[3]120052@uotechnology.edu.iq

1. INTRODUCTION

It is well known that the WSN has been considered in numerous fields of the modern life, such as smart buildings and emergency systems [1]. These applications provide the WSN with high importance and should be more reliable, flexible and extendable. These properties have been addressed by designing the software algorithms according to the software engineering techniques [2]. The fault tolerance method is one of the most important part in the software engineering that can guarantee the operation of a system with high reliability. To be more precise, the self-checking technique can be considered for checking the status of such system based on the readings with the time. The concepts of this technique are built on allocating upper and lower threshold values for right operation to any part of a system. These values are used for checking the validity of the system, in which the fault can be detected efficiently [3].

In general, the fault in WSN are occurred mainly in the nodes (sensors) [1]. These faults can be classified into hardware and software. These two classifications are also being under the permanent and temporary faults. As mentioned above, the self-checking approach is used for detecting the fault of nodes and the monitoring system sends a message for maintenance for solving the occurred problems [2]. Throughout the maintenance procedure, the system can cover this error in different ways depending on the design and status of the related conditions [3].

In this paper, a monitoring system for WSN is proposed based on software engineering techniques. The proposed method adopts the fault tolerance approach for managing the operation of nodes inside the WSN. In the fault detection part of the proposed method, the self-checking technique is considered with upper and lower threshold for each node included in a cluster. The designed WSN contains numerous clusters and each of which includes different nodes. At a time that the fault is occurred in a node, the proposed method takes the neighbor nodes reading to tackle the problem in an efficient way. The recovery operation is performed by calculating the average value of nearest neighbor nodes until the fault is solved. The proposed system is tested under different conditions and the achieved results prove the satisfactory in terms of accuracy and efficiency.

2. RELATED WORK

The applications of WSN and related supported systems attract the attentions of numerous researchers in distinct fields. Each group of them tackled a side of appeared problems, such as faults, data transmission. All these researches focused on introducing robust systems with high accuracy, reliability, flexibility and extendibility. Therefore, the concepts and approaches of software engineering were adopted.

In [4], the authors presented a developed management method for WSN based on software engineering rules. They classified the situations of WSN in terms of infrastructure and found the suitable adapting software. While the authors of [5] proposed a real time oil and gas monitoring system based on WSN using the software development as a part of software engineering. The presented system took care of different equipment, such as pumps, sensors and metering infrastructure. In addition, it managed the flow of oil and gas sequentially. In [6], a new framework for WSN was presented. The introduced framework was built based on software engineering challenges including flexibility and scalability. It worked as a middleware that could manage the data mining, energy consumption and self-organization. In addition, the authors of [7] performed a modeling framework for analyzing the features in terms of variation and commination. These features provided a complete view for the WSN based agriculture system. All proposed algorithms and software were designed under the rules of software engineering. In [8], a searching based on software engineering was implemented in WSN. The implemented method adopted the multi-objective algorithm to be solved using genetic method. It also considered the fixed obstacles in the searching process. The authors of [9] introduced hardware and software infrastructure for WSN based on the regulation of software engineering, particularly the software requirements. The focused on the receiver and how it described itself to the others nodes of WSN. The work was done in physical layer in real time situation. It is important to note that the test results of the literature research works were clear enough to prove the claim of authors.

At the other hand, the fault tolerance of software engineering was adopted in solving the problem of reliability. In [10], a survey on fault detection and the solutions were presented. It considered the relation between different types of faults in WSN nodes and the choosing of suitable fault detector as well as fault tolerance method. This relation could lead to the root of faulted node, in which the solution was presented. The authors of [11] proposed a management architecture of WSN based on fault tolerance approach. The proposed architecture tackled two sides of challenges inside WSN. The first one is the power consumption, while the other side is the fault tolerance to cover the self-organized network. This was to increase the reliability of the WSN and become fault tolerance. In [12], a new trend of guaranteeing the reliability of WSN was presented based on fault tolerance method. Both theory and applications were considered in the oriented research. In [13], a cluster head based fault tolerance system was proposed. The objectives of the proposed system were to keep eye on the WSN throughout the operation in terms of data transmission, mobility and fault occur.

The solutions were also provided to guarantee the reliability. The authors of [14] used an agent factor to propose a fault tolerance method for WSN. The platform of the proposed method worked across different levels including sensors, cluster heads and base stations. In [15], the researchers employed the fault tolerance method for reducing the redundancy and packet loss over the broken links (paths) of WSN and finding the solutions to keep the WSN in operation. Moreover, they supported the security of the data transmission using cryptography methods. The authors of [16] designed a software engineering based fault detection method for optimization method that was proposed. In [17], a list of the recent methods of fault detection and diagnosing was presented. It included a real survey on fault detection in WSN. The researchers of [18] proposed a fault detection based clustering network. The proposed method involved two sides; firstly, the cluster configuration, while the other is fault detection and recovery. In [19], a method of solving the problem of node isolation in AGR-MAC protocol was presented. The proposed method used fault tolerance technique for maintaining the isolation problem. The authors of [20] proposed a guaranteed algorithm of bioinformatic connectivity in WSN based on fault tolerance technique for ambulance system. In [21], a WSN based wildfire detection system was proposed based on self-organization and fault tolerance approaches.

At the other hand, a system based virtual biological community of nature living style has been proposed. The evolutionary game method was adopted to introduce a fault tolerance procedure for managing the problems of WSN [22]. In [23], a fault tolerance method was presented for solving the problem of cluster head failures. This was done by planning the solution based on virtual heads and the collected information. The researcher of [24] proposed a data transmission protocol in save situation using fault tolerance method. This protocol used the residual information regarding the links of WSN to build a plan for tolerant the faults. In [25], the cluster head selection method was proposed based on fault tolerance and fuzzy-logic approaches. The soft computing method provided the system with optimality in allocating the cluster head, in which the failure is covered well. In [26, 27], different methods for WSN structure and clustering methods have been proposed to solve numerous problem. They used software engineering regulations in building the proposed methods.

3. THE PROPOSED SYSTEM

As mentioned earlier, the proposed system includes two sides of work based on software engineering methods that are used for guarantee the reliability of the adopted WSN. The first one is related to detect the occurred fault in a node at the adopted WSN using the self-checking method. While the second one takes care of finding the solution of the detected fault using the available tolerance in finding the recovery values within the neighborhood area.

3.1. System structure

In this section, the general structure of the proposed system is explained as a block diagram shown in Figure 1. It is shown that the collected reading data from the underlying sensors of allocated WSN is entered to the fault detection unit. This unit is responsible on detecting the fault that can be occurred in a node using the self-checking approach. This approach is based on upper and lower threshold values for each sensor as a tolerance range. The received values that fall within the outage tolerance area is classified as a fault. If there is no detected fault, the data is sent to the system safely.

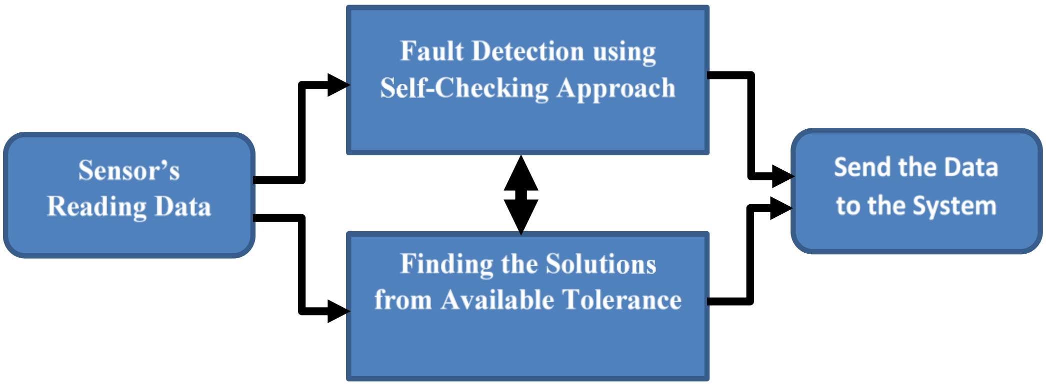

Figure 1. The proposed system structure

The detected faults are passed to the fault tolerance unit. This unit is responsible on finding the solution to the problem of fault using the available tolerances. The solution adopted in the proposed method uses the values of neighbor nodes with correct values for guessing the expected reading of the faulted sensor node. Whilst the processing of finding the solutions, a communication between fault detection unit and fault tolerance unit. The reason behind this communication is exchanging the information and correct any error can be happened in finding the tolerant solution. The results of fault detection and the tolerant solution is applied to send the corrected readings as well as the estimated reading to the system for continuing the process.

3.2. Proposed algorithm

It is well known that the software algorithm is necessary to be proposed for managing the fault detection and finding the solution. In this section, the proposed algorithm is presented as a flowchart, shown in Figure 2.

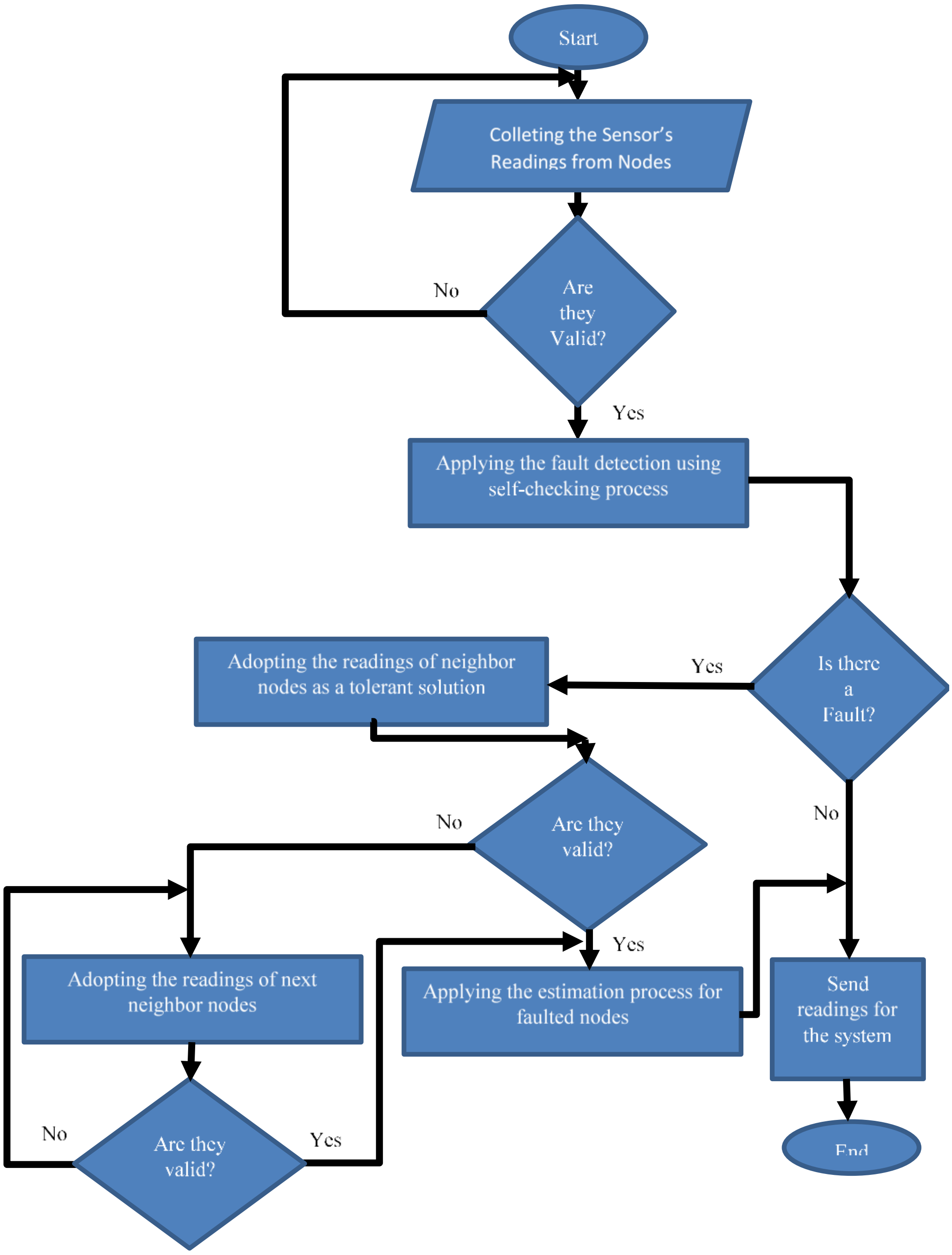

Figure 2. Flowchart of the proposed algorithm

Based on the flowchart of the proposed algorithm, the working procedure can be summarized in the following:

a. Collecting the readings of the sensor nodes in the adopted WSN.
b. Checking the validity of the received readings by the cluster head. In case the readings are not valid, a request is sent back to the nodes for providing another copy of readings.
c. The valid readings are checked for fault detection using the self-checking process based on the upper and lower threshold as explained mathematically in (1):

$$\gamma_{upper} \geq \delta(i,j) \geq \gamma_{lower} \quad (1)$$

Where γ_{upper} and γ_{lower} are the upper and lower threshold values. In case of fault occur, the proposed algorithm sends the information to the next step. Otherwise, the readings can be passed to the system.

It is important to highlight that the γ_{upper} and γ_{lower} values are selected depending on the type of used sensor as well as the surrounding conditions.

d. The faulted readings are passed to the fault tolerance unit. This unit adopts the valid readings of the close neighbour nodes. The most important condition is the validity of these reading as could be happened the even the close neighbour readings have faults as well according to the following equation:

$$\delta(\boldsymbol{i},\boldsymbol{j}) =, \begin{Bmatrix} \delta(i \pm k, j \pm k), & \gamma_{upper} \geq \delta(i \pm k, j \pm k) \geq \gamma_{lower} \\ \delta(i,j) \quad , & otherwise \end{Bmatrix} \tag{2}$$

Where $\boldsymbol{\delta(i,j)}$ is the instant reading of a node with location (i,j) in the covered area. $\boldsymbol{\gamma(i \pm k, j \pm k)}$ is the next neighbor readings, increased by $\boldsymbol{k} \in \boldsymbol{K}$ total WSN dimension, and k={1,2,3,…,K}.

e. It is noted from (2) hat the instant reading of a node can be replaced with the neighbour nodes in case they are valid. In order to increase the validity of estimation and the accuracy, the average value can be adopted amongst the valid close neighbours in the estimation process. Therefore, the estimated value $E_{\delta(i,j)}$ of reading of a node $\delta(\boldsymbol{i},\boldsymbol{j})$ is mathematically written as:

$$\boldsymbol{E}_{\delta(i,j)} = \frac{1}{N_{node}} \sum_{k=1}^{K} \delta(i \pm k, j \pm k) \tag{3}$$

Where $\boldsymbol{N_{node}}$ is the total number of adopted neighbor nodes to the underlying node $\delta(\boldsymbol{i},\boldsymbol{j})$.

f. Checking the estimated value of a node $\boldsymbol{E}_{\delta(i,j)}$ if it is within the upper and lower threshold vlues as:

$$\gamma_{upper} \geq \boldsymbol{E}_{\delta(i,j)} \geq \gamma_{lower} \tag{4}$$

g. Sending the valid values (sensors' readings) to the system for future processing.

4. RESULTS

In order to test the proposed method that has been explained in early sections, a simulation model is adopted. The proposed model includes a WSN with four clusters, each of which involves cluster head and six sensors, as shown in Figure 3. The cluster heads are connected to the device manager as a base station. Temperature sensors are adopted in this simulated model that can be changed to different types of sensors. This model is built using Visual Studio C# language and the interface presents the results in numerous color for more explanation. The green boxes represent the well working nodes, while the red boxes are the faulted nodes. Inside each box, the instant sensor reading is shown. The nodes are connected to the cluster head within each one individually.

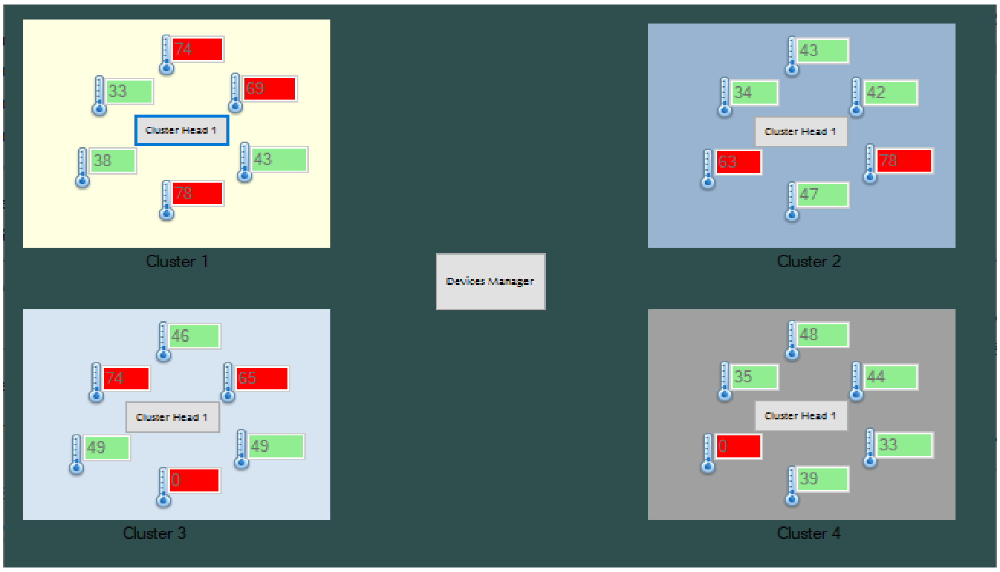

Figure 3. Simulated model interface, case study

Different case studies have been considered to test the system under changeable conditions. The case study, shown in Figure 3, explains that the cluster one has three faulted sensor nodes, mentioned in red color. In addition, cluster two involves two faulted sensors with red label. Moreover, cluster three includes a faulted sensor, while the fourth cluster points three sensor nodes to be faulted. Depending on the proposed algorithm and based on equations (1-4), the proposed method compensates the faulted readings with estimated values. It is important to note that the proposed algorithm amongst all clusters checks the readings of neighbor nodes if they are within the thresholds to be considered. Otherwise, it considers the next neighbor instead to guarantee the estimation with minimum error occur. The simulation parameters are built based on the upper and lower threshold of (35° c) and (55° c), sequentially, for a whole July month in Baghdad city at Iraq.

Figure 4 shows the same case study of Figure 3 after applying the proposed algorithm. I can be seen that the faulted readings of the sensors in all clusters are estimated based on the neighbors and labeled in blue color. As mentioned earlier, the results are measured for thirty days among July month. The performance of each sensor in all clusters can be measured accurately. Figure 5 shows the performance of sensor nodes in cluster one. The case here that there is no fault is detected, and all sensors are working well in cluster one. It is seen that the whole readings are within the validity range between upper and lower thresholds.

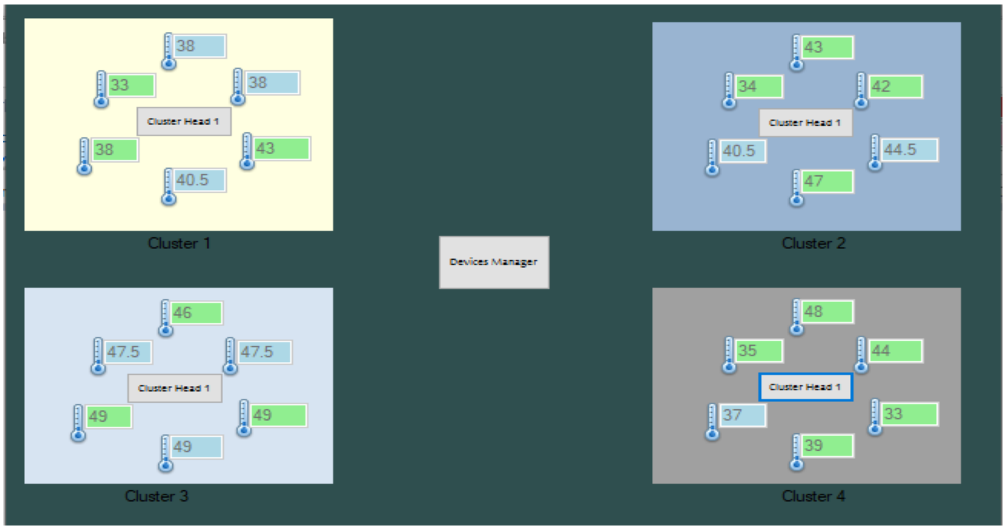

Figure 4. Case study of Figure 3 after applying the proposed algorithm

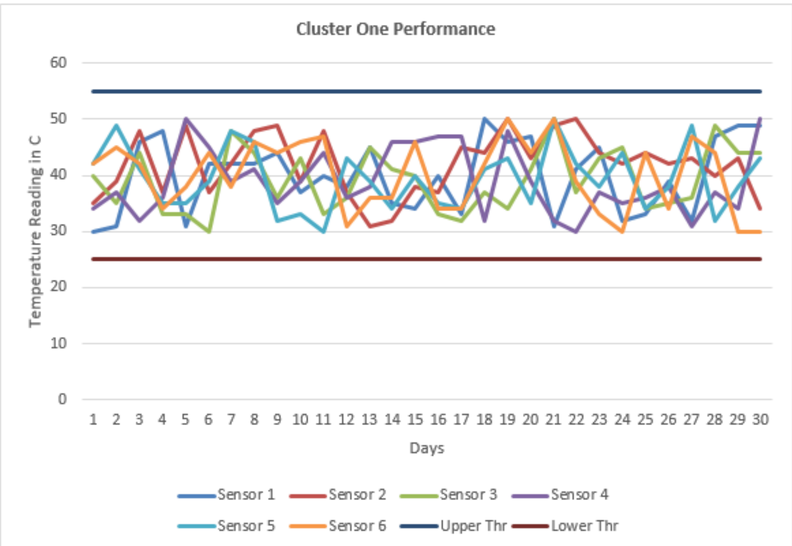

Figure 5. Cluster one performance in case of normal working

At the other hand, the same cluster one passes by worst conditions, in which there are three sensors suffer from faults. This can lead to changing in the readings of these sensors. These faults are detected using the proposed algorithm that is based on self-checking process. The detected faults are compensated by applying the proposed fault tolerance method based on the neighbor nodes as explained earlier. Figure 6 explains the readings of sensors inside cluster one, where the faulted readings exceed the upper threshold values. It is shown that the faults are started at day seven, twelve and fourteen, sequentially.

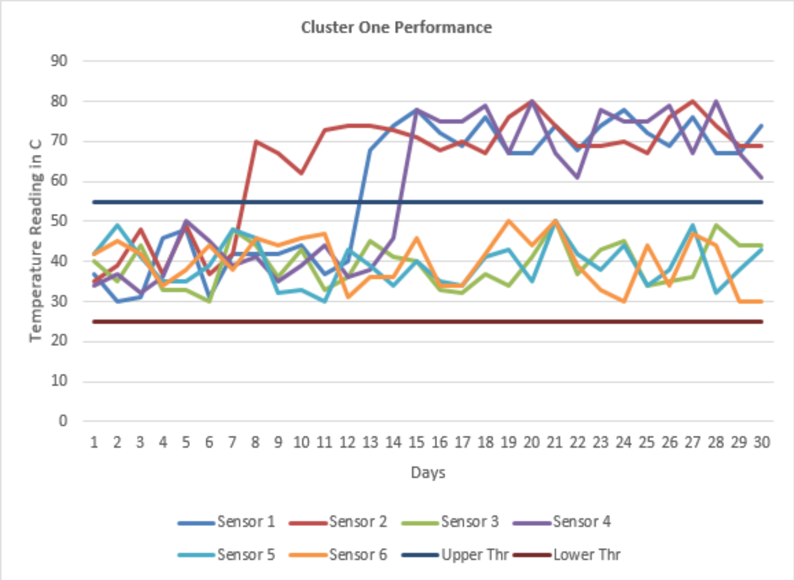

Figure 6. Cluster one performance in case of fault detection

Now the proposed method should solve this problem by applying the presented algorithms. This is done efficiently as shown in Figure 7. The faulted readings are fixed by estimating replaced values depending on the valid readings of neighbor nodes. To be more precise, sensor node number four in cluster one has been analyzed in Figure 8. It is shown that the estimated values of this sensor are within the validity range between thresholds with acceptable error.

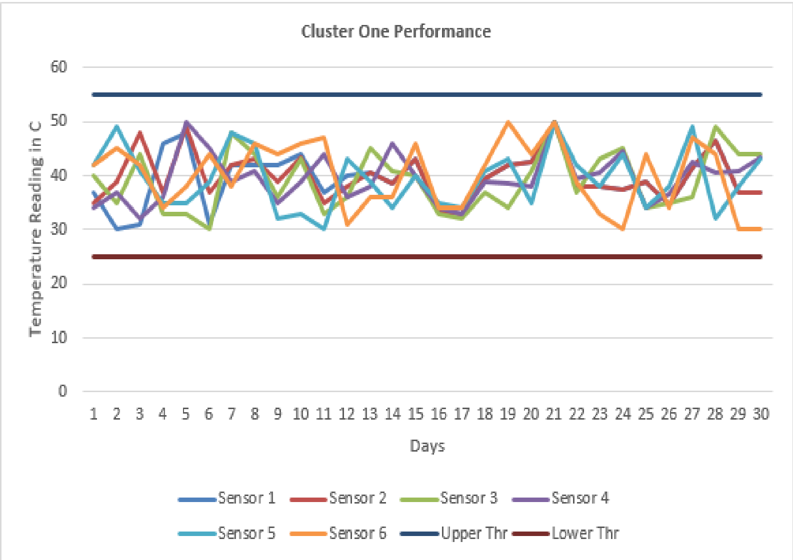

Figure 7. Cluster one performance in case of fixing faults

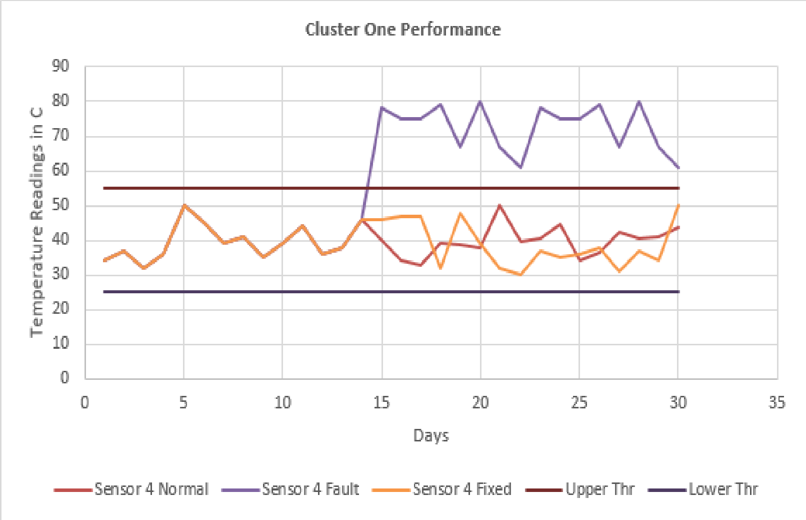

Figure 8. Sensor four analysis regarding normal, fault and fixed

Figure 9 illustrates the performance of the adopted WSN with four clusters. For each cluster, the average values of normal, faulted and fixed cases are evaluated according to the following equation:

$$Av_{\delta(i,j)}(z) = \frac{1}{N_{Total}} \sum_{i=1}^{I} \sum_{j=1}^{J} \delta(i,j) \quad (5)$$

where $Av_{\delta(i,j)}(z)$ is the average values of the cluster ***z***, and *z={1, 2, 3, …, Z}* and ***Z*** is the number of clusters within the WSN. N_{Total} is the number of all readings of sensors inside the cluster ***z***, represented as a matrix of ***(6*** X ***30)*** for six sensors and thirty days. ***I*** is considered as the number of sensors, which is six, while ***J*** represents the number of days (thirty days). These values are compared in this figure and shown that the estimated values are very close to the normal ones, while the faulted reading increased in notable way. This means that the proposed algorithm and method are working in efficient way.

Figure 9. The total cluster performance for normal, faulted and fixed cases

Table 1 shows the recorded readings of sensors over thirty days of July month for cluster three at the present of fault occur, as an example to the proposed system fault monitoring. It can be seen that the values of readings of the 6th sensor are increased sharply after the 10th day as the fault happened. In addition, the readings of sensor four decreased to become zeros and this is indicated as a fault as the values went down the lower threshold starting from day 11th. At the other hand, sensor two provides wrong readings after 23rd day, where the values are increased notably. The values, highlighted with yellow color, are recorded as faults due to the exceeding of upper and lower thresholds (25-55^{o}c), which are the maximum and minimum temperatures among this month.

Table 1. Cluster 3 sensors after fault (in sensor 2, 4, 6)

Day	Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5	Sensor 6
1.	36	43	31	38	34	47
2.	40	40	49	40	43	49
3.	44	37	31	34	31	48
4.	39	42	41	44	32	32
5.	47	32	36	49	34	38
6.	50	36	48	43	41	36
7.	46	48	41	49	50	34
8.	46	46	32	30	49	37
9.	32	33	31	45	40	43
10.	49	44	48	45	37	76
11.	39	37	40	0	39	70
12.	40	39	38	0	43	71
13.	34	38	39	0	31	76
14.	33	30	49	0	45	74
15.	39	43	34	0	35	71
16.	41	49	30	0	45	70
17.	47	42	43	0	38	75
18.	41	40	33	0	38	63
19.	41	50	49	0	40	78
20.	45	34	34	0	34	68
21.	47	45	38	0	40	68
22.	32	42	35	0	32	74
23.	46	65	49	0	49	74
24.	38	65	31	0	32	63
25.	45	59	34	0	31	60
26.	33	75	37	0	31	58
27.	35	66	30	0	47	74
28.	37	68	31	0	36	63
29.	34	64	50	0	40	77
30.	33	75	30	0	39	78

It is important to note that the proposed algorithm does not change the complexity of the WSN as there is no any addition to the hardware side. In addition, in fault appearance cases, the solution is produced until they are fixed. Therefore, the proposed algorithm consumed a sensible time to recover the errors, in which the real-time concepts are still satisfied.

5. CONCLUSION

This paper proposed a software engineering based fault tolerance method for fault detection and introducing a solution in WSN. The proposed method adopted the self-checking process for fault detection. In addition, the proposed algorithm considered the neighbor sensor nodes to estimate the faulted readings in efficient way. The average value of the close neighbors was adopted in the proposed algorithm to increase the accuracy of estimation as well as the reliability. In case of fault occur in the close neighbor node, the next valid one was considered. It is important to note that the robust response of the proposed method allowed the use in real-time system with high activity of fault tolerance. The proposed method was tested over different case studies at an assumed simulation model with different clusters in WSN. The obtained results showed the proved performance of the proposed method in terms of reliability and accuracy.

REFERENCES

[1] Hossam Mahmoud Ahmad, "Wireless Sensor Networks Concepts, Applications, Experimentation and Analysis," *Springer*, 2016.
[2] Ian Sommerville, "Software engineering," *10th Edition, Pearson Education, Inc*, 2017.
[3] Miikka Kuutilaa, Mika Mantylaa, Umar Farooqa, and Maelick Claes, "Time Pressure in Software Engineering: A Systematic Review," *Esliver*, 2020.
[4] V. V. Phoha and Shashi Phoha, "Situation-Aware Software Engineering for Sensor Networks," *2nd International Conference on Communication Systems Software and Middleware*, 2007.

[5] Ayuba J, Safwana H, and Abdulazeez Y, "Software Development of Integrated Wireless Sensor Networks For RealTime Monitoring of Oil and Gas Flow Rate Metering Infrastructure," *Journal of Information Technology & Software Engineering*, vol. 8, no. 2, pp. 1-10, 2018.

[6] Savitha S, S C Linga Reddy, Sanjay Chitnis, "Energy Efficient Clustering and Routing Optimization Model for Maximizing Lifetime of Wireless Sensor Network," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 5, 2020.

[7] Sridhar R., and N. Guruprasad, "Energy efficient chaotic whale optimization technique for data gathering in wireless sensor network," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 4, pp. 4176-4188, 2020.

[8] Abdusy Syarif, Abdelhafid Abouaissa, Lhassane Idoumghar, Achmad Kodar, Pascal Lorenz, "Search Based Software Engineering on Evolutionary Multi-Objective Approach," *IEEE- ICC Next-Generation Networking and Internet Symposium*, 2016.

[9] Aaditiya Venkat Kannan, "Hardware and Software Architecture of Wireless Sensor Networks," *Journal of Advances in Computer Networks*, vol. 2, no. 3, pp. 207-210, 2014.

[10] Emmanuel Effah, Ousmane Thiare, "Survey: Faults, Fault Detection and Fault Tolerance Techniques in Wireless Sensor Networks," *International Journal of Computer Science and Information Security*, 2018.

[11] Khadidja El Merraoui, Abdellaziz Ferdjouni, M'hamed Bounekhla, "Real time observer-based stator fault diagnosis for IM," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 1, pp. 210-222, 2020.

[12] Sushruta Mishra, Lambodar Jena, Aarti Pradhan, "Fault Tolerance in Wireless Sensor Networks," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 2, no. 10, pp. 146-153, 2014.

[13] Manasvi Mannan, and Shashi B. Rana, "Fault Tolerance in Wireless Sensor Network," *International Journal of Current Engineering and Technology*, vol. 5, no. 3, pp. 1785-1788, 2015.

[14] A.V. Sutagundar, Vidya S. Bennur, Anusha A. M. Bhanu K. N., "Agent Based Fault Tolerance in Wireless Sensor Networks," *International Conference on Inventive Computation Technologies (ICICT)*, 2016.

[15] H.S.Annapurna, and M.Siddappa, "Secure Data Aggregation with Fault Tolerance for Wireless Sensor Networks," *International Conference on Emerging Research in Electronics, Computer Science and Technology*, 2015.

[16] TaoChena, MiqingLi, and XinYaocb, "Standing on the shoulders of giants: Seeding search-based multi-objective optimization with prior knowledge for software service composition," *Journal of Information and Softwrae Technology*, vol. 114, pp. 155-175, 2019.

[17] Zeyu Zhang, Amjad Mehmood, Lei Shu, Zhiqiang Huo, Yu Zhang and Mithun Mukherjee, "A Survey on Fault Diagnosis in Wireless Sensor Networks," *IEEE Access*, vol. 8, pp. 11349-11364, 2018.

[18] Mehdi Nazari Cheraghlou, Ahmad Khadem-Zadeh and Majid Haghparast, "A New Fault-Tolerant Clustering Protocol for Wireless Sensor Networks," *International Journal of Security and Its Applications*, vol. 12, no. 4, pp. 29-40, 2018.

[19] Myeongseung Han, Eun-Sung Cho, Jun-hyuk Lee, and Intae Ryoo, "A Study on Fault Tolerance Technique for AGR-MAC Protocol based Sensor Networks," *International Conference on Information Networking (ICOIN)*, pp. 457-460, 2019.

[20] Asmaa Shaalan Abdul Munem, and Muayad Sadik Croock, "Smart Traffic Light Control System for Emergency Ambulance," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, vol. 5, no. 1, 2016.

[21] Felipe Taliar Giuntini, Delano Medeiros Beder and Jo´ Ueyama, "Exploiting Self-organization and Fault Tolerance in Wireless Sensor Networks: A Case Study on Wildfire Detection Application," *International Journal of Distributed Sensor Networks*, vol. 13, no. 4, pp. 1-16, 2017.

[22] Ricardo Villalón and Patrick G. Bridges, "Fault-Tolerant Wireless Sensor Networks using Evolutionary Games," PhD Thesis, The University of New Mexico, 2012.

[23] Jenn-Wei Lin, Pethuru Raj C., Meng-Chieh H., and Jia-Xin H., "Efficient Fault-Tolerant Routing in IoT Wireless Sensor Networks Based on Bipartite-Flow Graph Modeling," *IEEE Access*, vol. 7, pp. 14022-14034, 2019.

[24] L. Mottola, G. P. Picco, F. J. Oppermann, et al, "Simplifying the Integration of Wireless Sensor Networks into Business Processes," *IEEE Transaction on Software Engineering*, vol. 45, no. 6, pp. 576-596, 2019.

[25] Farnaz Pakdel 1, Mansour Esmaeilpour, "Fuzzy Logic Method for Enhancement Fault-Tolerant of Cluster Head in Wireless Sensor Networks Clustering," *TEM Journal*, vol. 5, no. 3, pp. 268-277, 2016.

[26] Evizal Abdul Kadir, Hitoshi Irie, Sri Listia Rosa, and Mahmod Othman, "Modelling of wireless sensor networks for detection land and forest fire hotspot," *TELKOMNIKA (Telecommunication, Computing, Electronics and Control)*, vol. 17, no. 6, pp. 2772-2781, 2019.

[27] Ngan Nguyen, Quoc Cuong Nguyen, and Minh Thuy Le, "A novel autonomous wireless sensor node for IoT applications," *TELKOMNIKA (Telecommunication, Computing, Electronics and Control)*, vol. 17, no. 5, pp. 2389-2399, 2019.